AN AUTOMATED COST UNCERTAINTY PROGRAM

THESIS

DALE N. FLETCHER
GM-13

AFIT/GCA/LAS/94S-4

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GCA/LAS/94S-4

AN AUTOMATED COST UNCERTAINTY PROGRAM

THESIS

DALE N. FLETCHER
GM-13

AFIT/GCA/LAS/94S-4

AFIT/GCA/LAS/94S-4

AN AUTOMATED COST UNCERTAINTY PROGRAM

THESIS

Presented to the Faculty of the Graduate School of Logistics and Acquisition Management

of the Air Force Institute of Technology

Air University

In Partial Fulfillment of the

Requirements for the Degree of

Master of Science in Cost Analysis

Dale N. Fletcher, B.S.

GM-13

September 1994

Approved for public release; distribution unlimited

Acknowledgements

I wish to thank MAJ Wendell Simpson and LTC Carl Templin, my thesis advisors for their guidance, assistance, encouragement and most of all their patience throughout this research effort. A special thanks to Dr. Dave Barr and Prof. Dan Reynolds for supplying their expertise in statistics and computer programming. I would also like to thank the 94 GCA class for their support. Most importantly, I would like to thank my wife Denise for her presence and understanding.

Dale N. Fletcher

Table of Contents

<u>List of Figures</u>

Abstract

The purpose of this research was to develop a menu driven, computer program that implemented heuristic methodologies for cost uncertainty analysis.

Cost uncertainties have been quantified using either simulation techniques or through heuristic means. Simulation techniques require time to run the required number of iterations. Current computer based heuristic methods do not check correlation matrices for consistency nor do they enable the analyst to perform querys.

This research resulted in the development of an automated heuristic cost uncertainty program which alleviates the limitations of current computer programs. The program contains a file editor for creating and modifying data files, a cost uncertainty module for computing decile level total system cost probabilities and a query function.

The FORTRAN computer program is completely menu driven for ease of use.

AN AUTOMATED COST UNCERTAINTY PROGRAM

I. Introduction

General Issue

The cost of weapon systems is estimated utilizing various techniques and/or methodologies. Parametric equations, analogous systems, and engineering estimates are common types of techniques employed (21:12). The estimates obtained from these techniques yield a singular value or point estimate. Garvey defines the point estimate as the cost derived for an element in the work breakdown structure which excludes the effect of uncertainties in the element's technical definition or cost estimation methodology (1:2;14:165). Typically these cost estimates do not assign nor address a level of confidence which can be placed in these values. By accepting these point estimates as the expected cost of a system, we are inferring an accuracy in these values that may not be warranted (21:1). As a result, the cost uncertainty accompanying these point estimates is ignored.

Weapon systems have exceeded their estimated values as a result of variations in system configuration, changes in quantities, unstable program schedules and fluctuations in labor and material costs (21:9). These factors are of major concern to cost analysts for it is within these areas that the majority of the program's uncertainty can be found

(8:1). These uncertainties often manifest themselves as negative impacts to a weapon system's cost.

The major sources of uncertainty can be grouped into two broad categories. These categories are: requirements uncertainty and cost-estimating uncertainty (12:1). Requirements uncertainty originates from changes in a system's specified configuration. System configuration denotes deviations from original specifications or assumptions regarding hardware characteristics and/or system operational concepts. Cost estimating uncertainty refers to variations in cost estimates of a system when the configuration of the system remains constant (8:1). It originates from errors in cost models and methodologies, technical definition uncertainties, economic uncertainties and uncertainties in the data (11:4;16:2-9).

At best, an estimate represents a snapshot of what the cost of a system would be if all the assumptions upon which it is based were simultaneously realized (16:9). Cost, however, is a variable quantity that depends upon a set of conditions, any of which can change over the life cycle of the system. To answer the question: What is the chance that a particular estimate is likely to be exceeded?, it is helpful to present an estimate as a probability distribution. This approach permits random variation to operate in the variables that determine a system's cost estimate.

A work breakdown structure (WBS) is a hierarchical system of subordinate-level cost elements that are directly related to activities that define a project under development or production (23:13). There exists a distribution of cost for each WBS cost element. Each cost element has an associated probability density function (PDF) encompassing its true estimated cost (14:163). The probability density function represents the distribution of probability for an event occurrence (26:6). The area under the curve defined by an element's PDF will equal one (30:1). Cost uncertainty analysis is applied to cost estimates through the cost element WBS.

The primary output or goal of cost uncertainty analysis is the probability distribution of total system cost. This is known as the cumulative probability distribution (16:12). It is formulated by accumulating the input probabilities (30:2). The cumulative probability distribution depicts the set of all possible outcomes for a specified degree of random variation in the cost of individual elements (16:12). Generating the cumulative probability distribution of total system cost provides important insights into issues such as: determining the change and the amount that a point estimate is likely to be exceeded, and isolating the cost variance drivers and contributing factors that account for the build-up of uncertainty (16:14). Generating the total system cost depends however, on the assumptions regarding the shape of

the individual cost element distributions as well as their interdependencies (26:6).

Cost uncertainties can be quantified either through heuristic means or by way of simulation (23:2). The most notable of the simulation approaches is the Monte Carlo technique. This method has been used extensively to simulate cost (32:VII-11) or schedule uncertainties (34:66-70). However, simulation techniques in general can become quite expensive with respect to the amount of resources required. This is due primarily to the computational time necessary to run the specified number of iterations (30-3).

Heuristic methods have been proposed as an alternative to simulation. Analytical methods assume a distribution for total system cost rather than the empirical approximation employed by Monte Carlo Simulation. One such method, developed by Phillip Young, is the Formal Risk Assessment of System Cost Estimates (FRISK) which employs a heuristic approach in estimating a system's cost risk (1). The most advanced of these analytical models is Paul Garvey and Audrey Taub's method for estimating cost and schedule uncertainties using joint probability distributions (15).

Specific Problem

At present, most heuristic methods for estimating cost uncertainty require a cost analyst to create their own implementation. For the more sophisticated models, the user must not only have the time to perform the tedious calculations but must also be fully versed in both

statistics and calculus and must have access to a mathematical solver like Mathcad® or Mathmatica®. Existing computer based implementations of heuristic methods such as FRISK, are not complete in that they assume a consistent correlation matrix (37). In addition, these software programs do not allow the analyst to perform querys. As a result, these alternatives to Monte Carlo simulation are restrictive in their capabilities.

Research Objective

The goal of this research is to develop a menu driven, computer program which will implement heuristic methodologies for cost uncertainty analysis. It is primarily because of the speed with which heuristic methodologies perform probability computations that this research has settled on developing an automated heuristic cost uncertainty model. This effort has centered around alleviating the limitations of current computer programs by instituting correlation consistency checking and incorporating a query capability. The following objectives will be accomplished in order to address the previously mentioned problem:

1. Identify and assess the usefulness of the different uncertainty methodologies as well as the different computer models currently available to the analyst.

2. Determine recommended approaches to designing software.

3. Develop a menu driven computer program which employs the heuristic approach in performing cost uncertainty analysis.

The model, titled the Automated Cost Uncertainty Program (ACUP), seeks to build on the tools and techniques already available to the analyst. ACUP incorporates the attributes of the previously mentioned models and adds the normal and lognormal distributions as added options for cost element input distributions. In addition to expanding the types of input distributions, the model incorporates a consistency check of the correlation matrix and provides the analyst with the added capability of querying the model for a probability given a cost estimate or alternatively, computing a cost requirement given a specific probability level.

Chapter two of this paper will cover the first objective. A review of uncertainty in cost analysis and current techniques and models will be accomplished.

Chapter three will review a recommended approach to designing computer software. Chapter four presents the program's design including a discussion of the algorithms.

Chapter five concludes this paper with a summary and an area for further study.

II. Literature Review

Overview

This chapter will present an overview of cost uncertainty analysis techniques. These techniques may be divided into two categories: simulation and heuristic (30:2).

Simulation Methods

Simulation methods generally use the Monte Carlo method to derive the total cost distribution by sampling the input distributions and then using convolution to obtain the shape of the total cost distribution. Convolution is a mathematical method of summing two or more statistically independent probability density functions (26:21).

The initial step in the simulation process requires the analyst to specify a probability distribution for each cost element. Most simulation programs provide the user with various choices of possible distributions. The Normal, Lognormal, Exponential and Weibull are examples of the types of distributions available (22:158-168). This method for obtaining the total cost distribution is straightforward when the cost elements to be simulated are independent. Each cost element realization is generated as a random draw from its specified distribution. This procedure is repeated typically 500 to 1,000 times in order to obtain a sample from the unknown distribution of cost (32:VII-11). After all cost elements have been generated, they are then

summed to obtain a single realization for the total cost (23:24-25). The frequency distribution of these outcomes is an approximation of the distribution of the cost of the total system (32:VII-11). However, the lack of an acceptable method for generating correlated, non-normal random variables with bounded domains (such as the Beta and Triangular distributions) is one major drawback (23:25).

Monte Carlo simulation has been established as an accurate method for cost uncertainty analysis. The advantage of the simulation method is that it allows more realism in the analysis (32:VII-11). The technique is accurate in that the total cost probability distribution function produced by a simulation is a close approximation of the 'true' total cost distribution (30:3). Performing a Monte Carlo simulation however, can be expensive with respect to the time required to conduct the simulation. Another drawback to performing simulations is the lack of an acceptable method for generating correlated non-normal random variables (23:25).

<u>Heuristic Methods</u>

Two representative heuristic models are Abramson and Young's Formal Risk Evaluation Methodology (FRISKEM) model and Garvey's Analytical Cost Probability (ACOP) model (1;14). Both of these models were developed as alternatives to simulation methods. The FRISKEM model does not allow the analyst to employ any distribution shapes other than triangular. The model assumes all lower level WBS cost

element distributions can be characterized by the same cost distribution shape. FRISKEM also assumes the total cost distribution is lognormally distributed (1). Alternatively, the ACOP model is flexible enough to accommodate all types of lower level cost element distributions (14). In contrast to FRISKEM, the ACOP model assumes the total cost distribution is normally distributed (14:167). Simpson and Grant investigated how the shape (normal or lognormal) of the total cost distribution impacts the total system cost. They showed that for uncertainty analyses containing more than ten cost elements, the normal distribution models the total cost distribution reasonably accurately (30:7-16). Both the FRISKEM and ACOP models accomplish quantification of dependencies among cost elements by incorporating a correlation matrix (1;14). However, the responsibility of assuring consistency among the pair-wise correlation values between lower level cost elements is left to the analyst.

The principle advantage to using an analytical technique is the rapidity with which this method derives its computations. Heuristic techniques on the other hand have been characterized as having one major deficiency in that the total cost distribution shape is initially assumed to be either normally or lognormally distributed (30:4).

III. Methodology

Introduction

This chapter will provide an overview of the process by which the Automated Cost Uncertainty Program (ACUP) was developed.

Software Design

Freeman and Wasserman state that increased effort in the earlier stages of software development will be reflected in a software product of high quality and more likely to fulfill the needs of its users (13:610). Emphasis was therefore placed on the initial planning of this research effort. The preliminary effort required the development of a software design methodology.

Software design is a structured process which delineates the specific phases of software development. The phases are analysis, functional specification, design, implementation, validation and evolution and are defined as follows:

> Analysis - a step concerned with understanding the problem and describing the activities, data, information flow, relationships and constraints of the problem; the typical result of the analysis phase is a requirements definition;
>
> Functional Specification - the process of going from the statement of the requirements to a description of the functions to be performed by the system to process the required data;
>
> Design - the process of devising the *internal* structure of the software to provide functions specified in the previous stage, resulting in a description of the system structure, the architecture of the system components, the

algorithms to be used, and the logical data structures;

Implementation - the production of executable code that realizes the design specified in the previous stage;

Validation - the process of assuring that *each* phase of the development process is of acceptable quality and is an accurate transformation from the previous phase;

Evolution - the ongoing modifications (repair, adaptation to new conditions, enhancement with new functions) to a system caused by new requirements and/or the discovery of errors in the current version of the system (13:611).

Analysis. Most cost uncertainty models are based on a Monte Carlo (simulation) method and furthermore assume that all cost elements are statistically independent (26:26). The existing automated heuristic models which allow for statistically dependent cost elements do not include correlation consistency checking nor a query capability (1). Therefore, an automated heuristic approach which incorporates both a means of checking correlation consistency and a query function can increase the productivity of the cost uncertainty user.

Functional Specification. Cost uncertainty methodologies rely on the definition of cost distributions for each cost element (26:6). To be compatible with existing cost uncertainty methodologies, the program must utilize similar data input conventions. Typically, these conventions require each cost element be defined in terms of a range of possible costs (minimum, most likely and maximum). The minimum cost corresponds to a value that

would be realized only under the most fortuitous circumstances while the maximum value reflects a pessimistic perspective (21:13). The most likely value represents the cost element's cost mode or point estimate (32:VII-5). In addition to estimating the range of possible costs, a shape is specified that describes the probability distribution of cost within the range of possible values (16:12). These distribution shapes can range from the simplistic uniform to the flexible beta (22).

Five cost element distribution shapes were chosen as allowable inputs to the program. The shapes selected were the uniform, triangular, beta, normal and lognormal distributions. The uniform distribution is appropriate in situations where the analyst is completely uncertain about a given cost except that it must be between some specified minimum and maximum values. The triangular distribution can be used when a most likely value can be ascertained along with a minimum and maximum and a piecewise linear density function. The beta distribution is useful because it can take a wide variety of shapes from negatively skewed to bell-shaped to positively skewed (32:VII-3). The normal and lognormal distributions are included as input distribution shapes because they represent more precisely uncertainty situations of symmetry and skewness respectively.

Minimum, most likely and maximum values can be used to compute the mean and variance for each cost element given any one of the previously mentioned distribution shapes.

However, in certain instances minimum and maximum values are not available. Rather, the selected distribution's mode and standard deviation are provided. The program must be able to accept data entries in the form of minimum/maximum values as well as in the form of a mode and standard deviation.

The final piece of information required by cost uncertainty models is a quantification of the dependencies among cost elements. Garvey, in his paper *A General Analytic Approach To System Cost Uncertainty Analysis*, cautions against assuming all cost elements are independent by stating:

> It is typically an oversimplification to assume that all of the engineering activities defined by the cost elements of a system are independent. Such assumptions can yield low estimates of a system's cost variance, and therefore, unrealistically small differences in cost between the tails of the probability distribution is likely to be seen (14:167).

Garvey further admonishes that, "correlation among the elements of cost can be an important contributor to the magnitude of the system cost variance (14:167)." Simpson and Grant add further credence to this position by noting the assumption of independent cost elements is difficult to defend (30:6). Therefore, in order to more accurately approximate total cost variance, the interdependencies between cost elements must be considered.

Existing cost uncertainty models allow for quantification of dependencies among cost elements in the form of a correlation matrix (1:1). However, these models do not check the correlation matrix for consistency. A

correlation matrix is considered to be consistent if the following criteria are met:

1. The correlation coefficients are consistent in sign;
2. The correlation coefficients are consistent in magnitude;
3. The cost element distribution shapes are consistent with the correlation coefficients (25:10).

Figure 1 portrays these relationships. It therefore becomes the responsibility of the user of any dependent cost uncertainty methodology, as noted by O'Hara, to specify internally consistent correlations (26:49). An erroneous total cost variance due to inconsistent correlation coefficients can be identified with the inclusion of a consistency check.

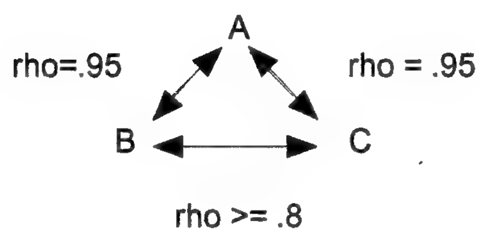
A major output of cost uncertainty analysis is the probability distribution (PDF) of total system cost (16:12). The total cost PDF provides two valuable pieces of information. A range in which the total system is likely to fall and an estimate of probabilities associated with possible values of total system cost (30:2). Generating the total cost PDF depends on the assumptions of cost dependency and the shape of the cost element distributions (26:6). Cost element dependencies can be addressed using correlation values. The availability of different input distribution types provides the analyst flexibility in the selection of

Consistency Check #1
Consistent in Sign

Permissible signs for rho (n=3)

A <-> B	A <-> C	B <-> C	
+	+	+	OK
+	+	-	NO
+	-	-	OK
-	-	-	NO

Consistency Check #2
Consistent in Sign



Consistency Check #3
Consistent in Sign

Rho between A and B is .95

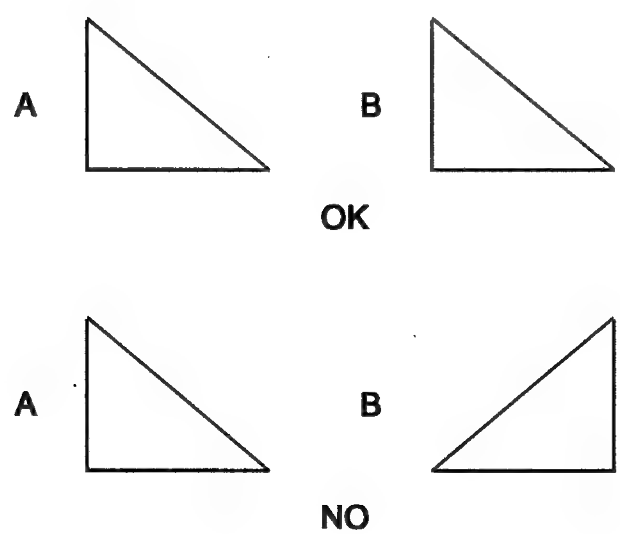


Figure 1. Correlation Consistency Checking

an appropriate distribution shape for each cost element. However, the shape of the total cost distribution has been purported to be normal as well as lognormal (1:1;21:17-25). It is beyond the scope of this paper to determine which (if any) total cost distribution shape is more correct. Simpson and Grant have addressed the issue of differing total cost distribution shapes. In their paper *An Investigation of Heuristic Methods for Cost Uncertainty Analysis*, they determined the normal distribution more accurately depicts uncertainty analyses containing ten or more elements. Because both the normal and lognormal distributions have been used to characterize the shape of the total cost distribution, the program must provide results expressed in terms of both distributions.

<u>Design</u>. The Automated Cost Uncertainty Program (ACUP) is a menu driven computer program comprised of separate and distinct subprograms each designed to perform specific tasks. These tasks include file editing, correlation consistency checking, cost uncertainty computations and the query routine. The modularity of the program facilitates coding and testing. Figure 2 displays the program organization and data flow (20:30-31).

<u>Implementation</u>. The program was compiled on a personal computer and will only run on a 80386 or better machine. The programming language used for ACUP is FORTRAN77. FORTRAN77 is a problem oriented language which enables users

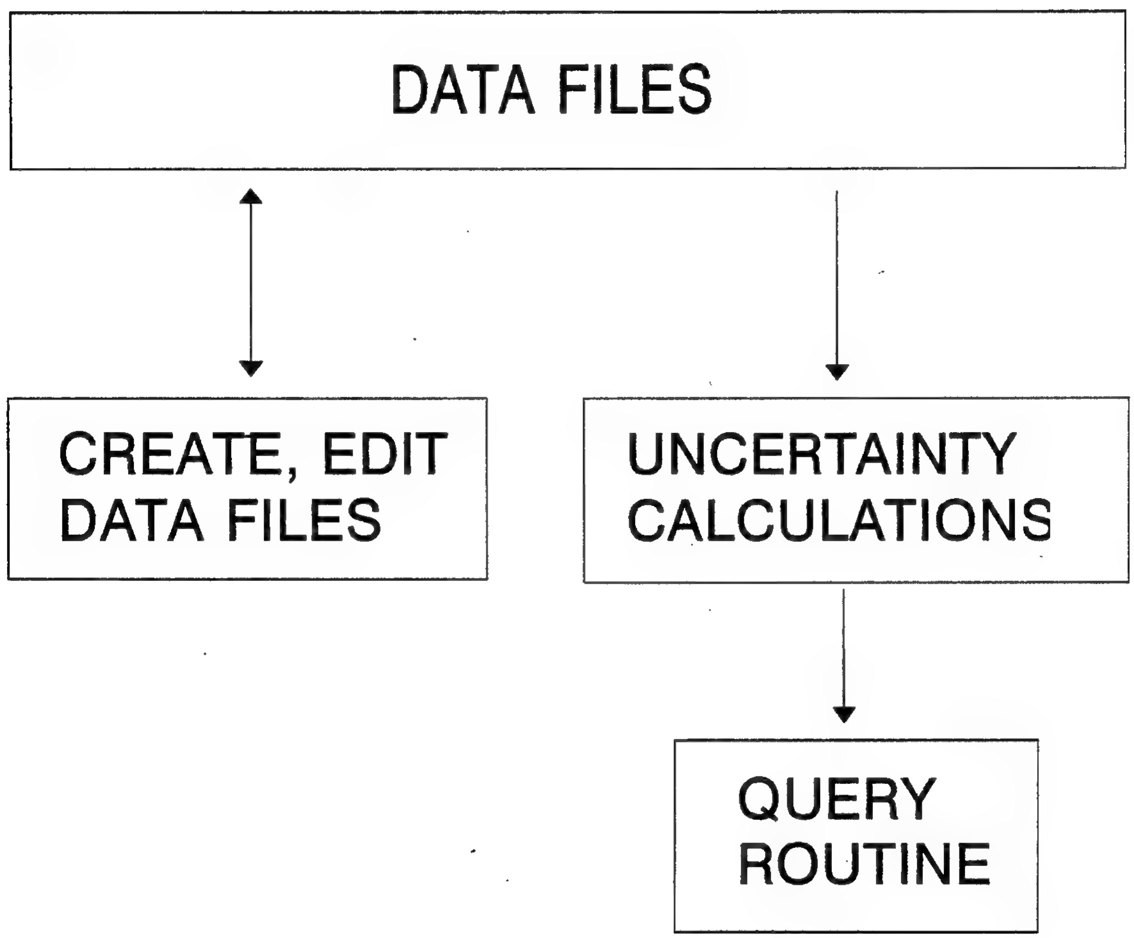


Figure 2. Program Data Flow

to write the solution of problems utilizing familiar mathematical expressions (4:7). The reasons for the selection of FORTRAN77 were due to its excellent mathematical capabilities and the accessibility of existing FORTRAN subroutines.

Three of the subroutines which perform essential computations were incorporated into the program (3:27). The following list specifies the three incorporated subroutines and gives a brief description of their functions.

Jacobi. Jacobi is a FORTRAN subroutine that is used to compute the eigenvalues and eigenvectors of a square matrix (33:112-114). ACUP uses this subprogram to check correlation matrices for consistency. The computed eigenvalues and eigenvectors were verified using test matrices (17:55-61).

ALNORM. This FORTRAN subroutine calculates the upper, or lower, tail area of the standardized normal curve corresponding to any given argument (2;6:186-188;9;18:374-375;19:424-427). ALNORM is used in the ACUP query routine to compute an uncertainty level (probability) given a total cost estimate.

PPND16. This last FORTRAN subroutine computes the percentage point of the standard normal distribution corresponding to a prescribed value of the lower tail area (35:477-484). PPND16 is used by ACUP as the counterpart to

ALNORM. This subroutine returns a total cost given an uncertainty level.

Values obtained from the subroutines ALNORM and PPND16 were compared to probability integral tables in order to verify their accuracy and precision (10:227;27:116-120).

Validation and Verification. As stated previously, the modularity of ACUP facilitated extensive testing. The program's modularity allowed for faster fault isolation. Program testing was accomplished in three stages: subprogram testing, integration testing and acceptance testing (28).

Subprogram Testing. In subprogram testing, individual program units were tested for correctness. This entailed entering controlled data into each subprogram unit and comparing the observed outputs with known results. As an example, test data (mode, min/max or standard deviation) alues were entered for each distribution type. The computed mean and variances were compared to known values obtained from a mathematical solver. The mathematical solver used for this thesis was Mathcad 5.0. In this way, the mean and variance computations for each of the five distribution types was verified for accuracy. Test matrices specifically designed for assessing the accuracy of algorithms was used to verify the correctness of the correlation consistency checking. This effort included entering both valid and inconsistent correlation values and verifying the resulting

computed eigenvalues. In addition, this insured that matrices declared inconsistent were indeed inconsistent and those matrices the program deemed valid were in fact valid. Finally, the accuracy of the integration subroutines used by the query routine to compute costs and probabilities was verified using integral tables which represent areas under the normal curve for values ranging from .0001 to .9999.

Integration Testing. Following successful subprogram testing the individual subprograms were progressively combined and tested. This type of testing provides confirmation that values passed between subprogram units maintain their integrity. Integration testing also exposes errors resulting from the improper specification of subprogram interrelationships. One major error that surfaced during this phase was the loss of significance resulting from the logarithmic calculations. This error was overcome by increasing the number of significant digits maintained by the program. Not until all subprograms were successfully combined and demonstrated to be error free was this phase considered complete.

Acceptance Testing. Due to time constraints and personnel availability, analysts were selected from the Air Force Institute of Technology's Graduate Cost Analysis Program to test the complete program. A program user's guide, three test cases and evaluation forms were made

available (see Appendix C). The test cases represented fictitious cost uncertainty problems.

Case #1 contained six dependent cost elements. The individual distributions for these cost elements was triangular. The range of allowable costs were expressed in terms of minimum/maximum values. This case was a duplicate of an uncertainty problem previously solved by the graduate students using simulation. It was included in the evaluation in order to give the test participants a tangible means of comparing the speed of heuristic techniques with that of simulation.

Case #2 contained nine cost elements. All five distribution types (uniform, triangular, normal, lognormal and beta) were represented in this case. The range of allowable costs were required to be entered either as minimum/maximum values or as a standard deviation. Unlike case #1 where all cost elements were dependent, several of the cost elements were designated as independent. Inclusion of non-correlated elements was used to insure the resulting correlation data file would not be affected by independent elements.

The third and last case combined the attributes of the first two cases with file editing requirements. A temporary file (*WORKFILE.CST*) containing twenty three independent cost elements was supplied to the participants. They were required to add, delete and modify cost element data within both the cost and correlation data files.

The analysts reported and documented any erroneous results using the supplied evaluation forms. The first testing sequence uncovered problems with the clarity of the program. Data requests by the program were in certain instances not clear to the analysts which in turn led to erroneous computations. Program termination due to incorrect data specification was another problem identified and corrected during this sequence. Once these errors were corrected, the test was then administered to a second group of analysts and the process repeated. The test participants were also instructed to defeat the program. During this sequence, problems such as entering correlation values outside of the acceptable range (-1 to +1) led to program execution errors. The program was honed and refined based largely on user inputs and suggestions.

Evolution. Modularity was deliberately designed into the program to aide in verification and validation. The relative independence of subprogram units also facilitates the program's upgrading. Individual subprogram elements may be altered without compromising the entire program. Entire subprograms may be replaced as long as the original program interfaces are maintained. The adaptability of the program to changing conditions will insure its continued utility.

IV. Program Design

Introduction

Chapter four consists of three sections. Each section represents an overview of the program's three main functions corresponding to file editing, cost uncertainty and the query function.

Program Structure

The computer program contains the following three application modules:

1. File Management: The file editor allows the analyst to create, edit and view the cost and correlation data files.

2. Cost Uncertainty Analysis: This module will compute the total cost uncertainty probabilities (deciles) depicting a normal total system distribution and a lognormal total system distribution.

3. Query: This capability is a subset of the Cost Uncertainty Analysis module and can only be accessed from that module. It allows the user to compute the probability associated with an estimate. The analyst is alternatively able to compute the total cost associated with a specified probability level. These computations can be applied to either the normal or lognormal probability distributions.

The first two modules can be accessed from the main menu. As was stated previously, the query function can only be accessed via the cost uncertainty analysis module. The

following subsections discuss the formulas and algorithms representing the foundation of the preceding three modules.

File Management. The first requirement of the file editor is to create a data file. The number of cost elements making up a work breakdown structure (WBS) can vary considerably. Some of the larger weapon system cost estimates have been observed to contain close to two hundred separate cost elements. The program was therefore designed to accommodate up to two hundred cost elements. This upper bound on the number of cost elements can be readily increased if necessary. This can be accomplished by increasing the size of the parameter value within the source code to a larger value (see Appendix B for source code). The parameter statement is used to size the arrays which hold the various data items.

The cost data is entered in the following format: cost element name, estimated cost (mode), distribution type (uniform, triangular, normal, lognormal and beta), either minimum/ maximum values, or a standard deviation, and an indication of dependence. As the information is entered, it is echoed onto the screen for verification. Once all the data has been entered for a particular cost cell, the entries along with the computed mean and variance are displayed at the bottom of the screen. After completion of the cost data input, the pairwise correlations must be entered. For each element the analyst previously annotated as being dependent, the program requests the following

inputs: correlated with which element(s) and the value or strength of the relationship. When the cost and the correlation information has been successfully entered, the files are saved on disk under a filename supplied by the analyst.

The edit routine allows the analyst to view either the cost data file or the correlation data file. The data within these files can then be edited as required. Options available to the user while in the edit routine include modifying existing cost elements, inserting additional elements and deleting elements. All editing functions are contained within the program and do not depend on the operating system of the host computer.

Cost Uncertainty Analysis. This section will discuss the formulas and algorithms used to compute the cost element mean and variance, check the correlation matrix for consistency and compute the total cost probabilities.

As part of the data file creation routine, the analyst is required to supply a point estimate (cost mode), minimum/maximum value or standard deviation, distribution shape and an indication of dependency for each cost element. The cost element's mean and variance are computed based on the input values and the type of probability distribution selected. Depending on the shape of the each cost element's distribution, the mean and variance for the individual elements are computed using the following equations:

Uniform Distribution. The mean and variation of the uniform distribution are computed using the following formulas:

$$\mu = \frac{\min + \max}{2} \tag{1}$$

$$\sigma^2 = \frac{(\min + \max)^2}{12} \tag{2}$$

where μ = mean
σ^2 = variance
min = minimum value
max = maximum value

Triangular Distribution.

$$\mu = \frac{\min + mode + \max}{3} \tag{3}$$

$$\sigma^2 = \frac{\min(\min - mode) + \max(\max - \min) + mode(mode - \max)}{18} \tag{4}$$

where μ = mean
σ^2 = variance
min = minimum value
max = maximum value
mode = point estimate

Beta Distribution. The beta distribution is specified as:

$$f(y) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)(h-1)^{\alpha+\beta-1}}(y-1)^{(\alpha-1)}(h-1)^{(\beta-1)} \tag{5}$$

where $\alpha>0$ and $\beta>0$ are shape parameters, l is the lower bound and h is the upper bound of the distribution (23:9). However, because α and β may not always be available and to maintain consistency with the input conventions of the other four distributions, PERT beta is used in its place. This methodology approximates the mean and variance of the beta

distribution using the parameters minimum, maximum, and mode (32:VII-11).

$$\mu = \frac{\min + 4mode + \max}{6} \tag{6}$$

$$\sigma^2 = \frac{(\max - \min)^2}{36} \tag{7}$$

where μ = mean
σ^2 = variance
min = minimum value
max = maximum value (32:VII-5)

Normal Distribution. A cost element with a normal probability distribution will accept a cost range expressed in terms of either a mode and standard deviation (σ) or minimum/maximum values. If the mode and standard deviation option is selected, the mean and the variance are computed as follows:

$$\mu = mode \tag{8}$$

$$\sigma^2 = \sigma * \sigma \tag{9}$$

Devore states that 99.7% of the values in any normal population lie within three standard deviations of the mean (7:150). Therefore the minimum and maximum values for the normal distribution can be approximated using the following equation:

$$\min = \mu - 3\sigma \tag{10}$$

$$\max = \mu + 3\sigma \tag{11}$$

If the range of the possible cost values for the normal distribution are entered in the form of minimum and maximum values, the following equation is used to approximate the variance.

$$\sigma^2 = \left(\frac{\max - \min}{6}\right)^2 \tag{12}$$

This approximation represents $\pm\, 3\sigma$ from the mean.

The normal distribution is symmetric with respect to its mean. Therefore the supplied minimum and maximum values should be equi-distant from the cost mode. If the minimum and maximum values do not describe symmetry about the cost mode, the program adjusts one of the endpoints. The program will compare the distance between the mode and each of the endpoints. The greater of the two distances is used as the new minimum or maximum value. This adjustment reflects a conservative estimate rather than an optimistic estimate.

Lognormal Distribution. A cost element with a lognormal probability distribution will accept a cost range expressed in terms of either a mode and standard deviation or as minimum/maximum values. If the mode ($mode_x$) and standard deviation (σ_y) option is selected, the analyst must insure that σ_y represents the standard deviation of the underlying normal distribution. This requirement stems from the fact that the standard deviation (σ_x) of a lognormal distribution does not lend itself to a closed form solution

when the mean (μ_x) is not available. Given the $mode_x$ and σ_y, the mean and variance of the underlying normal distribution is computed using the following equations:

$$\mu_y = \ln(mode_x) + \sigma_y^2 \tag{13}$$

$$\sigma_y^2 = \sigma_y * \sigma_y \tag{14}$$

Again, μ_y and σ_y^2 represent the mean and variance of the underlying normal distribution. The mean (μ_x) and variance (σ_x^2) of the lognormal distribution can then be computed using the equations:

$$\mu_x = e^{(\mu_y + \frac{\sigma_y^2}{2})} \tag{15}$$

$$\sigma_x^2 = e^{(2\mu_y + \sigma_y^2)} * (e^{\sigma_y^2} - 1) \tag{16}$$

Approximations for the minimum and maximum values are obtained using the fact that six standard deviations represent 99.7% of the values of a normal population.

$$\min_y = \mu_y - 3\sigma_y \tag{17}$$

$$\max_y = \mu_y + 3\sigma_y \tag{18}$$

Min_y and max_y represent the upper and lower values of the underlying normal distributions. These values are transformed to the values for the lognormal using:

$$min_x = e^{min_y} \tag{19}$$

$$max_x = e^{max_y} \tag{20}$$

As was the case for the normal distribution, the mean and variance of the lognormal distribution can be determined from the analyst supplied minimum and maximum values. These values are first transformed into the minimum and maximum values of its underlying normal distribution.

$$min_y = \ln(min_x) \tag{21}$$

$$max_y = \ln(max_x) \tag{22}$$

Since the cost element input distribution is lognormal, the probability density function is normally distributed in log space (7:167). Therefore, the mean (μ_y) and variance (σ_y^2) can be approximated using the same methodology as was used for the normal distribution.

$$\mu_y = \frac{max_y + min_y}{2} \tag{23}$$

$$\sigma_y^2 = \left(\frac{max_y - min_y}{6}\right)^2 \tag{24}$$

These values are then transformed to reflect the values of the lognormal distribution using the equations 15 and 16.

Correlation Consistency. Contained in O'Hara's review of cost risk models is a methodology for determining the validity of a correlation matrix (26). The logic is as follows:

> A square matrix C is said to be positive definite if it satisfies:
> (a) $C = C^H$, i.e., C is a Hermitian matrix (26:33).
> (b) $x^T Cx > 0$ for all $x \neq 0$ (5:431-435;24:554).
>
> A Hermitian matrix C is positive definite (positive semi-definite) if and only if all eigenvalues of C are positive (non-negative) (26:33;36:24-26).

Searle states that a correlation matrix is non-negative definite (21:348;26:33). Searle also notes that all symmetric matrices are a subset of Hermitian matrices (21:342;26:33). O'Hara concludes his observation by stating that the test for valid correlation matrices is to calculate the eigenvalues of the correlation matrix. The eigenvalues of a real symmetric matrix are real (5:411). If all eigenvalues are non-negative, the correlation matrix is valid (26:34).

The methodology for checking the validity of correlation matrices is contained within the Automated Cost Uncertainty Program. The program's algorithms used in determining consistency were tested using matrices specifically designed for testing computer computations (17;31:18).

The program computes the eigenvalues of the correlation matrices using the previously mentioned Jacobi FORTRAN subroutine. If any of the eigenvalues are less than zero, the correlation matrix is declared to be inconsistent.

<u>Total Cost Distribution</u>. The total cost probability distribution function (PDF) can be described by three pieces of information. The location (mean) of the PDF, the dispersion (variance) of the PDF and the shape of the PDF (30:4). If the lower level cost elements are independent the summation of their means and variances will result in the total cost mean and total cost variance.

$$\mu_{total} = \sum_{i=1} \mu_i \tag{25}$$

$$\sigma^2_{total} = \sum_{i=1}^{n} \sigma^2_i \tag{26}$$

If however, any of the lower level cost elements are dependent, it becomes necessary to quantify the relationship between these elements. This is accomplished by adding the covariance between cost elements to the total cost variance. Equation 27 shows the two components of the total cost variance. The first term represents the cost element variance. The second term captures the pair-wise interdependencies between cost elements.

$$\sigma^2_{total} = \sum_{i=1} \sigma^2_i + 2\sum_{i=2} \sum_{j=1} \sigma_{ij} \tag{27}$$

The covariance (σ_{ij}) can be estimated using the correlation coefficient (ρ_{ij}).

$$\sigma_{ij} = \rho_{ij}\sigma_i\sigma_j \tag{28}$$

(7:204)

The total cost mean for the dependent case is computed using equation 25. The correlation between cost elements does not affect the mean of the cost elements.

The mean of the total cost PDF (μ_{total}) and the variance of the total cost PDF (σ^2_{total}) are used to compute the percentiles in 10% increments (including 95% and 99%) for both normal and lognormal total cost distribution shapes. The equations used for the normal and lognormal total cost distributions are described in the following sections.

<u>Normally Distributed</u>. The calculation for the normal total cost distribution is as follows:

$$Probability(\%) = \mu_{total} + z_p * \sigma_{total} \tag{29}$$

Here, z_p represents the 100th percentile of the standard normal distribution. If z_p is the desired percentile for the standard normal distribution, the desired percentile for the normal distribution is then z standard deviations from the mean (7:151).

Lognormally Distributed. For the lognormal total cost distribution, the mean (μ_x) and variance (σ_x^2) are expressed in normal space and must be transformed into log-space using the following equations:

$$\mu_y = \frac{1}{2} * \ln\left(\frac{\mu_x^4}{\mu_x^2 + \sigma_x^2}\right) \tag{30}$$

$$\sigma_y^2 = \ln\left(\frac{\mu_x^2+\sigma_x^2}{\mu_x^2}\right) \qquad (1{:}4) \tag{31}$$

The total cost probabilities can then be expressed in normal space by using the following equation:

$$Probability(\%) = e^{(\mu_{total} + z_p * \sigma_{total})} \tag{32}$$

Query Function.

The query function is incorporated into the program to facilitate "what-if" exercises. It allows the analyst to enter a cost uncertainty level and obtain an associated total cost. Conversely, the analyst can provide a total cost value and obtain the related uncertainty level. These two options are made possible by the inclusion of the two previously mentioned FORTRAN subroutines, ALNORM and PPND16. These subroutines compute the area under the normal curve (probability) and compute the value (total cost) equated to a designated area under the curve.

V. Conclusion

Summary

The Automated Cost Uncertainty Program enables the analyst to employ heuristic methods in solving cost uncertainty problems. The program allows the analyst to compute probabilities associated with independent or dependent cost elements exhibiting either normal or lognormal total cost distributions. The program incorporates two capabilities previously not included in other automated heuristic models. The first capability enables the analyst to compute probabilities associated with different costs and the costs associated with different uncertainty levels. The second performs a consistency check of the correlation matrix.

The menu driven features of the Automated Cost Uncertainty Program enable the analyst to fully utilize the program's capabilities.

Areas for Future Research

ACUP allows the analyst to specify the range of possible costs for each cost element by entering the element's cost mode and either minimum/maximum values or a standard deviation. Currently, using the lognormal distribution requires input of the standard deviation of the underlying normal distribution in order to compute the mean and variance. A method allowing the analyst to enter the

The program employs heuristic methods to compute cost uncertainties. This same methodology can be employed to develop a program which would address schedule uncertainties. Together with the ACUP program, it would be possible to automate Garvey and Taub's method for estimating cost and schedule uncertainty using joint probability distributions (15).

The Automated Cost Uncertainty Program

USER'S GUIDE

by

Dale N. Fletcher
ASA(RD&A)

TABLE OF CONTENTS

INTRODUCTION

The major output of cost uncertainty analysis is the probability distribution of total system cost. This is known as the cumulative probability distribution. It depicts the set of all possible outcomes for a specified degree of random variation in the cost of individual elements. Generating the cumulative probability distribution of total system cost provides important insights into issues such as: determining the change and the amount that a point estimate is likely to be exceeded, and isolating the cost variance drivers and contributing factors that account for the build-up of risk.

Cost and schedule uncertainties can be quantified either through analytical means or by way of simulation. The most notable of these simulated approaches is the Monte Carlo technique. This method has been used extensively to simulate cost or schedule uncertainties. However, simulation techniques in general can become quite expensive with respect to the amount of resources required. This is due primarily to the computational time necessary to run the specified number of iterations.

Analytical methods have been proposed as an alternative to simulation. Analytical methods assume a distribution for total system cost rather than the empirical approximation employed by Monte Carlo Simulation. The advantage of analytical methods is that cost uncertainty can be estimated very quickly. Unfortunately, most analytical techniques require the analyst to either perform complex hand calculations or construct advanced computer spreadsheets in order to derive cost or schedule uncertainties.

OVERVIEW

The Automated Cost Uncertainty Program is comprised of two modules. The first is labeled the File Management Module. This module is used to create, edit and print the cost and correlation data files. The second, entitled the Cost Uncertainty Module is used to compute the associated probabilities. This last module contains a query function which facilitates what-if scenarios.

This program is designed to compute cost uncertainties utilizing the faster analytical techniques. The analyst is required to supply cost information to include the number of cost elements, the cost element's point estimate, the shape of the probability distribution (Uniform, Triangular, Normal, Lognormal or Beta) and the necessary distribution parameters. The resulting cost uncertainties are expressed as probabilities at the decile level and include 95% and 99%. The query function will compute the probability associated with a point estimate or alternatively a point estimate given an uncertainty level.

PART I GETTING STARTED

System Requirements

The Automated Cost Uncertainty Program is a DOS based program which is self contained on one 3.5" computer diskette. The program is designed to run on an 80386 or 80486-based computer. There are two files required to run the program which are contained on the accompanying diskette, UNCTY.EXE and DOSXMSF.EXE.

Program Installation

It is not necessary to install the program on a hard disk. The program may be run directly from the program diskette. However, the program may be copied to a hardisk using the supplied instalation program. To install the program on the hard disk, change to the drive containing the program diskette and enter the command INSTALL. The install procedure will create a directory on the hard drive labeled C:\UNCTY, and copy in the necessary files.

PART II FILE MANAGEMENT MODULE

File management is the first selection found under the program's *Main Menu*. This option enables access to the program's file editor. It is through the file management procedure the analyst is able to build as well as modify the program's required data files.

```
MAIN MENU

1)  File Management
2)  Cost Uncertainty
3)  Quit Program

    Enter Selection: 1
```

In order to perform a cost uncertainty analysis, the program requires two data files. A cost data file (file extension *.CST*) and a correlation data file (file extension *.COR*). To construct these files the user must first begin by selecting the option *Create Data Files* under the *File Management* menu.

```
FILE MANAGEMENT

1)  Create Data Files
2)  Edit Existing File
3)  Print Data Files
4)  Return to Main Menu

    Enter Selection:  1
```

Creating Cost Data Files

Selecting *Create Data Files* will allow the user to initially create a new cost and correlation data file. The program begins the data file creation procedure by first prompting the user for a filename (the file extension is not required). This filename will serve as the name for both the cost data file (filename.CST) as well as the correlation data file (filename.COR). The next prompt appearing on the screen will request the user to supply the number of cost elements the cost data file will contain. The program can accommodate between 1 and 200 distinct cost elements.

```
                         CREATE DATA FILE

Enter Filename:  COSTFILE

Enter number of cost elements: 4
```

Upon selecting *Create Data Files*, assigning a filename and indicating the desired number of cost elements, the next step is to supply the data for each cost element. This data includes the cost element name, the minimum/maximum values bounding the point estimate, the shape of the probability distribution, and an indication of whether the cost element is correlated with any of the other cost elements. The following screen will facilitate the data input process.

```
                          COST UNCERTAINTY
                             Data Input
   Element #
       3       CES Name                     Sys T&E

               Estimated Cost (mode)        986

               Distribution Shape           N
               (U,T,N,L,B)

       Select input type  1) Min/Max  2) Std Dev  1

                          Minimum          860
                          Maximum          1671
                Correlated? (Y/N)           Y

                      Is this correct? (Y/N) Y

  NAME         MIN      COST      MAX   DISTR    MEAN       VAR
  Sys T&E     301.0    986.0    1671.0    N      986.0    52136.1

                      Press [RETURN] to Continue
```

The available distribution shapes are the Uniform, Triangular, Normal, Lognormal and Beta (represented by U,T,N,L,B respectively). If either the normal or lognormal distribution shapes are selected, the analyst is given two alternative means of describing the shape parameters. Either minimum/maximum values may be entered or the distribution's

standard deviation may be entered. Again, this method is only available for the normal or lognormal distributions. The standard deviation is then translated into representative minimum and maximum values[1]. The user must insure the minimum and maximum values used for the normal distribution describe symmetry about the estimated cost mode.[2] Maximum, minimum and the cost mode figures may be entered with values as large as $9 billion. Once the data is entered and deemed correct, the inputs along with the computed mean and variance are displayed at the bottom of the screen. The last input required on the *Data Input* screen is an indication of whether or not this cost element is correlated with any other cost element.

Creating Correlation Data Files

The correlation data file contains the pairwise correlation coefficients (ρ-values) associated with the previously created cost data file. The *Create Correlation Matrix* screen follows directly after the cost input screen once all cost data information has been correctly entered.[3]

```
                    CREATE CORRELATION MATRIX

   Cost Element Number:  3             Cell Name:  Sys T&E

Current Correlations
 1  Init Spare    .70
 2  Data          .83
 4  Training      .88
                        Add correlation? (Y/N)  Y

                   Correlated With Element:     5

                   Correlation Coefficient:     -.79

 Cost        Cell       Correlated With      Cell        Correlation
Element      Name       Cost Element         Name        Coefficient
   3       Sys T&E           5              SE/PM           -.79

                      Is this correct? (Y/N)
```

This screen will display the cost element identified as being dependent during the data input process. The analyst is then prompted for the number of the cost element this cell is correlated with as well as the value of the correlation coefficient. The values for the correlation coefficient must range between -1 and +1. Values outside this interval will be

---

[1]See Appendix A for equations which translate σ to min/max values.

[2]NOTE: The normal distribution is symmetric with respect to the estimated cost mode. Therefore, if the input values used for the minimum and maximum do not describe a symmetric shape, the program will adjust either the minimum or maximum value to insure symmetry.

[3]If all cost elements are independent, this screen will not appear.

rejected. It is possible to stipulate that cost element '3' is correlated with element '2' as well as stating element '2' is correlated with element '3'. However, if the values entered are not equal, the previously entered correlation coefficient is displayed and the user is asked whether the existing value is to be replaced.

Currently

Cost Element	Cell Name	Correlated With Cost Element	Cell Name	Correlation Coefficient
4	Training	3	Training	.92

Replace? (Y/N)

An affirmative response will cause the previous value to be overwritten. Otherwise this value is ignored. The correlation input procedure for each dependent element will be terminated once the user indicates there are no more correlations to be added. After correlation values have been entered for each cost element previously designated as dependent during the data input procedure, the correlation information for all cost elements is then displayed on the screen.

CORRELATION VALUES

Element Number	Cell Name		
1	Init Spare	correlated with:	
		2 Data	.75
		3 Sys T&E	.70
2	Data	correlated with:	
		3 Sys T&E	.83
3	Sys T&E	correlated with:	
		4 Training	.88
		5 SE/PM	-.79

Press [RETURN] to Continue

Following the display of the correlation values, the validity of the correlation matrix is determined. This is to insure the relationships among the correlated cost elements are consistent. This procedure is transparent to the analyst. It requires the calculation of the eigenvalues of the correlation matrix. If all eigenvalues are non-negative the matrix is considered valid. If on the other hand any of the eigenvalues are negative, the matrix will fail the test and the analyst will be informed the matrix is invalid. The analyst must then rebuild the correlation matrix. This concludes the procedures used for the creation of the cost and correlation data files.

Editing Data Files

Option '2' of the *File Management* menu enables the user to add to, delete from or modify previously created cost and correlation data files.

```
FILE MANAGEMENT

1)  Create Data Files
2)  Edit Existing File
3)  Print Data Files
4)  Return to Main Menu

    Enter Selection:  2
```

Following the selection of *Edit Existing File*, the program prompts for the name of the datafile to be modified. It is not necessary nor is it required to identify the accompanying correlation data file as both files are retrieved simultaneously. The following screen will appear once the name of the file to be edited has been entered.

```
EDIT DATA FILE

1)  Edit Data File
2)  Save File
3)  Return to File Management

    Enter Selection: 1
```

To begin the file editor function, select *Edit Data File*. Choosing this option will display the cost data file to be modified and allow access to the various editing options.

```
                              EDIT COST DATA             costfile.CST

   Name               Min         Mode         Max       Distribution
1  Init Spare        1676.0      1942.0      2453.0           U
2  Data              3469.2      4029.7      5287.6           T
3  Sys T&E            301.1       986.0      1671.0           N
4  Training           366.7       576.2       963.9           L
5  SE/PM               10.9       287.2       402.5           B

 1) Insert   2) Delete   3) Modify   4) Next Page   5) Return  1
Insert after element number: 2
```

Inserting Cost Elements

Selecting option '1' will enable the analyst to add cost elements to the existing data. The user is prompted for the point of insertion. Although an element may not be added to the beginning of the data (as element number 1) , an element may be added to the end of the

list (as element number 6). Inserting a cost element is similar to the procedure used when creating a cost data file.

```
                              INSERT

   Element #
       3         CES Name                  Software

                 Estimated Cost (mode)     681

                 Distribution Shape        N
                 (U,T,N,L,B)

        Select input type  1) Min/Max  2) Std Dev  1

                          Minimum          371
                          Maximum          1011
                  Correlated? (Y/N)         N

                      Is this correct? (Y/N) Y

NAME          MIN       COST       MAX    DISTR     MEAN       VAR
Software     371.0     681.0     1011.0     N       681.0    11377.8

                    Press [RETURN] to Continue
```

As was the case when initially creating a cost data file, if the cost element is classified as dependent, the correlation data input screen will appear. The element is then added to the correlation data file. If, as in the example above the cost element is independent, this step is omitted and the analyst will be returned to the *EDIT COST DATA* screen.

Deleting Cost Elements

Option number '2' enables the user to delete a cost element. An element designated for deletion will only be removed from the cost data if the user supplies confirmation. Transparent to the user, the cost element is also deleted from the correlation data file.

```
                          EDIT COST DATA               costfile.CST

   Name                 Min         Mode        Max       Distribution
1  Init Spare         1676.0      1942.0      2453.0           U
2  Data               3469.2      4029.7      5287.6           T
3  Sys T&E             301.1       986.0      1671.0           N
4  Training            366.7       576.2       963.9           L
5  SE/PM                10.9       287.2       402.5           B

 1) Insert   2) Delete   3) Modify   4) Next Page   5) Return  2

Enter CES number to delete:  3
Delete CES number:     3  Sys T&E
    Confirm (Y/N):  Y
```

Modifying Cost Elements

Option number '3' of the *EDIT COST DATA* screen is used to change the data for each cost element. All data previously entered may be altered using this selection. The following screen will appear once the modify function has been invoked.

```
                         REPLACE

Element #     CES Name: Sys T&E           Replace (Y/N) Y
    3             Name: Testing

   Estimate (mode):             986.0     Replace (Y/N) Y
              Mode: 792

      Distribution:                 N     Replace (Y/N) Y
 Enter (U,T,N,L,B): B

     Minimum Value:             301.0     Replace (Y/N) Y
               Min: 639

     Maximum Value:            1011.0     Replace (Y/N) N

 Correlated? (Y/N):               Yes     Replace (Y/N) Y
       Enter (Y/N): N
```

In the above example it will be noted the cost element was changed from correlated to un-correlated. The accompanying correlation data will be modified to reflect this alteration. Transparent to the user all correlation values associated with the representative cost element (in this case Sys T&E) will be deleted. The *EDIT COST DATA* screen will reappear once the modification process is completed.

```
                          EDIT COST DATA              costfile.CST

   Name             Min         Mode         Max        Distribution
1  Init Spare     1676.0      1942.0      2453.0             U
2  Data           3469.2      4029.7      5287.6             T
3  Sys T&E         301.1       986.0      1671.0             N
4  Training        366.7       576.2       963.9             L
5  SE/PM            10.9       287.2       402.5             B

 1) Insert    2) Delete    3) Modify    4) Next Page    5) Return
```

The final two selections available to the user on this screen are *Next Page* and *Return*. Ten cost elements are displayed on one screen. *Next Page* will display the remaining cost elements. If the *Return* option is selected the program re-checks the correlation matrix for consistency and returns the user to the *EDIT DATA FILE* menu.

```
                EDIT DATA FILE

        1)  Edit Data File
        2)  Save File
        3)  Return to File Management

            Enter Selection: 2
```

All changes made to the cost and correlation data files to this point are temporary. Not until the user elects to save the file will the data become permanent.

Printing Data Files

To obtain a hardcopy of the data files, the analyst may select the *Print Data Files* option found under the *FILE MANAGEMENT* menu. This selection will enable the analyst to print the cost and correlation data contained in the file identified by the user.

```
                FILE MANAGEMENT

        1)  Create Data Files
        2)  Edit Existing File
        3)  Print Data Files
        4)  Return to Main Menu

            Enter Selection:  3
```

PART III COST UNCERTAINTY MODULE

Probability Output Report

The output report presents the total cost mean and standard deviation as well as the resulting total cost uncertainties associated with previously created cost and correlation data files. The report may be accessed via the second option under the *MAIN MENU*.

```
                MAIN MENU

        1)  File Management
        2)  Cost Uncertainty
        3)  Quit Program

            Enter Selection: 2
```

The output report displays the resulting total cost uncertainty expressed as probabilities at the decile level to include the 95% and 99% levels. In addition, the total cost mean and total cost standard deviation are displayed.

```
                     PROBABILITIES                   OLDFILE.CST
               Mean                     :       221.7
               Standard Deviation :              31.1

                          Normal               Log Normal
          Percentile    Value (est)           Value (est)
              10%          181.9                 183.6
              20%          195.5                 195.2
              30%          205.4                 204.0
              40%          213.8                 211.9
              50%          221.7                 219.5
              60%          229.5                 227.4
              70%          238.0                 236.2
              80%          247.8                 246.9
              90%          261.5                 262.5
              95%          272.7                 276.1
              99%          293.9                 303.6

              1) Query        2) Return to Main Menu  1
              3) Normal                  4) Lognormal  3
              5) Cost -> Prob       6) Prob -> Cost   5
              Enter Estimate:  250
                             81.92%
```

The Query Function

The example above shows the probability output screen. The report presents the different levels of the total cost probability distributed either normally or lognormally. At the bottom of the output report listed under option '1' is the query function. This function allows the user to perform what-if scenarios based on the currently referenced data files. The user may select either a normal or lognormal total cost distribution. Option '5' enables the user to obtain an uncertainty level (expressed as a probability) for a particular cost value.[4] In the previous example the query function was used to compute the probability associated with a cost of $250. Alternatively, option '6' of the query function may be used to obtain the cost required to attain a specific probability level. The probability inputs are restricted to the interval defined by .01% and 99.99% inclusive.

Finally, the user may terminate program execution by selecting option '3' of the *Main Menu*.

```
                    MAIN MENU

              1)   File Management
              2)   Cost Uncertainty
              3)   Quit Program

                   Enter Selection: 3
```

[4]See Appendix A for the methodologies used to compute the probabilities and costs using the query function.

APPENDIX A
Distribution Parameters

Uniform Distribution

Inputs: Estimated Cost (mode)
Minimum Value (min)
Maximum Value (max)

Compute: Cost Mean (μ)
Cost Variance (σ^2)

$$\mu = \frac{\min + \max}{2} \qquad \sigma^2 = \frac{(\max - \min)^2}{12}$$

Triangular Distribution

Inputs: Estimated Cost (mode)
Minimum Value (min)
Maximum Value (max)

Compute: Cost Mean (μ)
Cost Variance (σ^2)

$$\mu = \frac{\min + \mathrm{mode} + \max}{3}$$

$$\sigma^2 = \frac{\min \cdot (\min - \mathrm{mode}) + \max \cdot (\max - \min) + \mathrm{mode} \cdot (\mathrm{mode} - \max)}{18}$$

APPENDIX A
Distribution Parameters

Normal Distribution (method #1)

Inputs: Estimated Cost (mode)
Minimum Value (min)
Maximum Value (max)

Compute: Cost Mean (μ)
Cost Variance (σ^2)

$$\mu = \text{mode}$$

$$\sigma^2 = \left(\frac{\max - \min}{6}\right)^2$$

Note: Approximately 99.7% lie within three standard deviations of μ. This equation is used as an approximation of σ^2 for a normal distribution.

Normal Distribution (method #2)

Inputs: Estimated Cost (mode)
Standard Deviation (σ)

Compute: Minimum Value (min)
Maximum Value (max)
Cost Mean (μ)
Cost Variance (σ^2)

$$\mu = \text{mode} \qquad \sigma^2 = \sigma \cdot \sigma$$

$$\min = \mu - 3 \cdot \sigma \qquad \max = \mu + 3 \cdot \sigma$$

APPENDIX A
Distribution Parameters

Lognormal Distribution (method #1)

```
Inputs:   Estimated Cost   (mode_x)
          Minimum Value    (min_x)
          Maximum Value    (max_x)
```

Compute: Cost Mean (μ_x)
Cost Variance (σ_x^2)

$$\min_y = \ln(\min_x) \qquad \max_y = \mathrm{lin}(\max_x)$$

$$\mu_y = \frac{\max_y + \min_y}{2} \qquad \sigma_y^2 = \left(\frac{\max_y - \min_y}{6}\right)^2$$

Approximation of the σ^2 for a normal distribution.

$$\mu_x = e^{\mu_y + \frac{\sigma_y^2}{2}} \qquad \sigma_x^2 = e^{2\cdot\mu_y + \sigma_y^2} \cdot \left(e^{\sigma_y^2} - 1\right)$$

Lognormal Distribution (method #2)

Inputs: Estimated Cost ($mode_x$)
Standard Deviation (σ_y)

Compute: Minimum Value (min_x)
Maximum Value (max_x)
Cost Mean (μ_x)
Cost Variance (σ_x^2)

$$\mu_y = \ln(\mathrm{mode}_x) + \sigma_y^2$$

$$\min_y = \mu_y - 3\cdot\sigma_y \qquad \max_y = \mu_y + 3\cdot\sigma_y$$

$$\min_x = e^{\min_y} \qquad \max_x = e^{\max_y}$$

$$\mu_x = e^{\mu_y + \frac{\sigma_y^2}{2}} \qquad \sigma_x^2 = e^{2\cdot\mu_y + \sigma_y^2} \cdot \left(e^{\sigma_y^2} - 1\right)$$

Beta Distribution

```
Inputs:   Estimated Cost   (mode)
          Minimum Value   (min)
          Maximum Value   (max)

Compute: Cost Mean  (μ)
         Cost Variance  (σ2)
```

$$\mu = \frac{\min + 4 \cdot \text{mode} + \max}{6} \qquad \sigma^2 = \frac{(\max - \min)^2}{36}$$

```
      INCLUDE  'FGRAPH.FI'
      INCLUDE  'FGRAPH.FD'
C
      CALL CLEARSCREEN( $GCLEARSCREEN )
      CALL TITLE
      CALL INTRO
      CALL MENU
      END
C
C     Title Page
C
      SUBROUTINE TITLE
      PRINT 1
      PRINT 2
      PRINT 3
      PRINT 4
      PRINT 1
      PRINT 1
      PRINT 7
      PRINT 8
      PRINT 9
      PRINT 10
      PRINT 11
      PRINT 1
      PRINT 9
      PRINT 13
1     FORMAT ('0')
2     FORMAT (' ',31X,'The Automated')
3     FORMAT (' ',35X,'Cost')
4     FORMAT (' ',28X,'Uncertainty Program')
7     FORMAT (' ',32X,'Version 1.0')
8     FORMAT ('0',31X,'November 1994')
9     FORMAT (' ')
10    FORMAT ('0',36X,'by')
11    FORMAT ('0',30X,'Dale N. Fletcher')
13    FORMAT ('0',25X,'Press [RETURN] to Continue',\)
      READ (*,*)
      CALL CLEARSCREEN( $GCLEARSCREEN )
      END
C
C     Introduction Page
C
      SUBROUTINE INTRO
      WRITE (*,1)
      WRITE (*,4)
      WRITE (*,5)
      WRITE (*,6)
      WRITE (*,7)
      WRITE (*,8)
      WRITE (*,9)
      WRITE (*,10)
      WRITE (*,11)
      WRITE (*,12)
```

```
      WRITE (*,13)
      WRITE (*,14)
      WRITE (*,15)
      WRITE (*,16)
      WRITE (*,17)
      WRITE (*,18)
      WRITE (*,19)
      WRITE (*,20)
      WRITE (*,21)
      WRITE (*,3)
      READ (*,*)
1     FORMAT (' ',32X,'INTRODUCTION')
2     FORMAT (' ')
3     FORMAT ('0',26X,'Press [RETURN] to Continue',\)
4     FORMAT ('0',15X,'This program employs an analytical technique to')
5     FORMAT (' ',10X,
     +'analyze cost uncertainty.  The program is comprised of')
6     FORMAT (' ',10X,
     +'two modules.  The first module is used to create and edit')
7     FORMAT (' ',10X,
     +'the cost uncertainty data files while the second module')
8     FORMAT (' ',10X,
     +'is used to compute the probabilities.  The second module')
9     FORMAT (' ',10X,
     +'includes a query capability used to calculate uncertain-')
10    FORMAT (' ',10X,
     +'ties given either a cost estimate or a desired probability')
11    FORMAT (' ',10X,'level.')
12    FORMAT ('0',15X,
     +'The program requires the user to supply cost')
13    FORMAT (' ',10X,
     +'information to include the number of cost elements, the')
14    FORMAT (' ',10X,
     +'cost element point estimate, the shape of the probability')
15    FORMAT (' ',10X,
     +'distribution (Uniform, Triangular, Normal, Lognormal or')
16    FORMAT (' ',10X,
     +'Beta) and the necessary distribution parameters.')
17    FORMAT ('0',15X,
     +'The program presents the resulting cost uncertain-')
18    FORMAT (' ',10X,
     +'ties expressed as probabilities at the decile level to')
19    FORMAT (' ',10X,
     +'include 95 and 99%.  In addition, the program will')
20    FORMAT (' ',10X,
     +'compute the probability associated with a point estimate')
21    FORMAT (' ',10X,
     +'or a point estimate given an uncertainty level.')
      CALL CLEARSCREEN( $GCLEARSCREEN )
      END
C
C     Main Menu
C
      SUBROUTINE MENU
10    CALL CLEARSCREEN( $GCLEARSCREEN )
```

```
      PRINT 11
      PRINT 11
      WRITE (*,13)
      WRITE (*,14)
      WRITE (*,15)
C     WRITE (*,16)
C     WRITE (*,17)
      WRITE (*,18)
      PRINT 12
      WRITE (*,19)
      READ (*,*,ERR=10) SELECT
        IF (SELECT .EQ. 1) THEN
          CALL CLEARSCREEN( $GCLEARSCREEN )
          CALL FILMGT
        ELSE IF (SELECT .EQ. 2) THEN
          CALL CLEARSCREEN( $GCLEARSCREEN )
          CALL CSTUNC
        ELSE IF (SELECT .EQ. 3) THEN
          CALL CLEARSCREEN( $GCLEARSCREEN )
          CALL TERM
        ELSE
          CALL CLEARSCREEN( $GCLEARSCREEN )
          GOTO 10
        END IF
11    FORMAT ('0')
12    FORMAT (' ')
13    FORMAT ('0',34X,'MAIN MENU')
14    FORMAT ('0',26X,'1)   File Management')
15    FORMAT (' ',26X,'2)   Cost Uncertainty')
18    FORMAT (' ',26X,'3)   Quit Program')
19    FORMAT ('0',31X,'Enter Selection:  ',\)
      END
C
C     File Management Menu
C
      SUBROUTINE FILMGT
      PARAMETER (NP=200)
      DIMENSION MIN(NP),COST(NP),MAX(NP),DISTR(NP),CORMAT(NP,NP),
     +          NAME(NP),CORR(NP),STDDEV(NP),COUNT(NP)
      REAL COST,MIN,MAX,CORMAT,STDDEV
      INTEGER COUNT,CORR
      CHARACTER FLNM*8, DISTR*1, NAME*10
100   CALL CLEARSCREEN( $GCLEARSCREEN )
      I   = 0
      LINE= 0
      PRINT 2
      PRINT 2
      WRITE (*,53)
      WRITE (*,54)
      WRITE (*,55)
      WRITE (*,56)
      WRITE (*,57)
      PRINT 1
      WRITE (*,58)
      READ (*,*,ERR=100) SELECT
```

```
      IF (SELECT .EQ. 1) THEN
        CALL CLEARSCREEN( $GCLEARSCREEN )
        CALL CBUILD
      ELSE IF (SELECT .EQ. 2) THEN
        CALL CLEARSCREEN( $GCLEARSCREEN )
        CALL EDIT
C     Print data files
      ELSE IF (SELECT .EQ. 3) THEN
        CALL CLEARSCREEN( $GCLEARSCREEN )
        PRINT 2
        PRINT 2
        WRITE (*, '(1x,a\)') 'Enter name of file to print:  '
        READ (*,'(A)')FLNM
        OPEN (UNIT=1,FILE=FLNM(:LEN_TRIM(FLNM))//'.CST',
     +        STATUS='OLD',ACCESS='SEQUENTIAL',ERR=105)
        GOTO 110
105     CALL CLEARSCREEN( $GCLEARSCREEN )
        PRINT 1
        PRINT 2
        PRINT 2
        WRITE (*,60)
        PRINT 1
        PRINT 2
        PRINT 99
        READ (*,*)
        GOTO 100
110     OPEN (UNIT=2,FILE='LPT1')
        WRITE (2,8) FLNM(:LEN_TRIM(FLNM))
        WRITE (2,2)
        WRITE (2,5)
        WRITE (2,6)
          DO WHILE (.NOT. EOF(1))
            I=I+1
            READ (1,4) COUNT(I),NAME(I),MIN(I),COST(I),MAX(I),
     +                 DISTR(I),CORR(I),STDDEV(1)
            WRITE (2,7) I,NAME(I),MIN(I),COST(I),MAX(I),
     +                  DISTR(I)
            LINE=LINE+1
              IF (LINE.EQ.50) THEN
                WRITE (2,3)
                WRITE (2,19) FLNM(:LEN_TRIM(FLNM))
                WRITE (2,2)
                WRITE (2,5)
                WRITE (2,6)
                LINE=0
              END IF
          END DO
        CLOSE (UNIT=1,STATUS='KEEP')
        WRITE (2,3)
        OPEN (UNIT=3,FILE=FLNM(:LEN_TRIM(FLNM))//'.COR',
     +        STATUS='OLD',ACCESS='SEQUENTIAL',ERR=100)
          DO 115 L=1,I
            DO 120 M=1,I
              READ (3,9) CORMAT(L,M)
120         CONTINUE
```

```
115        CONTINUE
          CLOSE (UNIT=3,STATUS='KEEP')
          OPEN (UNIT=2,FILE='LPT1')
          LINE = 5
          WRITE (2,10) FLNM(:LEN_TRIM(FLNM))
          WRITE (2,1)
          WRITE (2,11)
          WRITE (2,12)
          WRITE (2,13)
            DO 210 L=1,I-1
              DO 220 K=L+1,I
                IF (CORMAT(L,K).NE.0) THEN
                  SUM=SUM+1
                END IF
220           CONTINUE
              IF (SUM.EQ.0) GOTO 210
              WRITE (2,14) L,NAME(L)
                DO 230 M=L+1,I
                  IF (CORMAT(L,M).EQ.0) GOTO 230
                  WRITE (2,15) M,NAME(M),CORMAT(L,M)
                  LINE=LINE+1
                  IF (LINE.EQ.46) THEN
                    WRITE (2,3)
                    WRITE (2,17) FLNM(:LEN_TRIM(FLNM))
                    PRINT 1
                    WRITE (2,11)
                    WRITE (2,12)
                    WRITE (2,13)
                    LINE=0
                  END IF
230             CONTINUE
              WRITE (2,16)
              LINE=LINE+2
              SUM = 0
210         CONTINUE
          WRITE (2,3)
          CLOSE (UNIT=2)
          GOTO 100
        ELSE IF (SELECT .EQ. 4) THEN
          CALL CLEARSCREEN( $GCLEARSCREEN )
          CALL MENU
        ELSE
          CALL CLEARSCREEN( $GCLEARSCREEN )
          GOTO 100
        END IF
1     FORMAT (' ')
2     FORMAT ('0')
3     FORMAT ('1')
4     FORMAT (I3,A10,3F15.4,A1,I1,F8.5)
5     FORMAT (' ',11X,'NAME',16X,'MIN',8X,'MODE',9X,'MAX'
     +          ,5X,'Distribution')
6     FORMAT (' ',11X,'----',16X,'---',8X,'----',9X,'---'
     +          ,5X,'------------')
7     FORMAT (6X,I3,3X,A10,1X,3F12.1,10X,A1)
8     FORMAT ('0',39X,'COST DATA',15X,A8,'.CST')
```

```
9       FORMAT (F5.2)
10      FORMAT ('0',30X,'CORRELATION VALUES',15X,A8,'.COR')
11      FORMAT (' ',13X,'Element',5X,'Cell')
12      FORMAT (' ',13X,'Number',6X,'Name')
13      FORMAT (' ',13X,'------',6X,'----')
14      FORMAT (' ',15X,I3,7X,A10,2X,'correlated with:')
15      FORMAT (' ',39X,I3,2X,A10,2X,F5.2)
16      FORMAT (' ',13X,'----------------------------------------------')
17      FORMAT ('0',30X,'Correlation Values',15X,A8,'.COR')
18      FORMAT (' ',29X,'Coefficient     ',\)
19      FORMAT ('0',33X,'Cost Data',15X,A8,'.CST')
53      FORMAT ('0',31X,'FILE MANAGEMENT')
54      FORMAT ('0',26X,'1)  Create Data Files')
55      FORMAT (' ',26X,'2)  Edit Existing Files')
56      FORMAT (' ',26X,'3)  Print Data Files')
57      FORMAT (' ',26X,'4)  Return to Main Menu')
58      FORMAT ('0',30X,'Enter Selection:  ',\)
60      FORMAT ('0',29X,'File Does Not Exist')
99      FORMAT ('0',26X,'Press [RETURN] to Continue',\)
        END
C
C       Create Cost and Correlation Data Files
C
        SUBROUTINE CBUILD
        PARAMETER (NP=200)
        DIMENSION COST(NP),MIN(NP),MAX(NP),DISTR(NP),CMEAN(NP),RHO(NP),
     +            CVAR(NP),NAME(NP),CORMAT(NP,NP),CORR(NP),TEST(NP,NP),
     +            STDDEV(NP),V(NP,NP),D(NP)
        REAL COST,MIN,MAX,TCMEAN,TCVAR,TCOST,RHO
        DOUBLE PRECISION LMIN,LMEAN,LMAX,LVAR,STDDEV,CMEAN,CVAR
        INTEGER I,COUNT,WITH,CORR,CORSUM
        CHARACTER FLNM*8, DISTR*1, NAME*10, CHOICE*1
100     PRINT 22
        PRINT 1
        WRITE (*, '(1x,a\)') 'Enter Filename: '
        READ (*,'(A)') FLNM
        OPEN (UNIT=0, FILE = FLNM(:LEN_TRIM(FLNM))//'.CST',
     +      ACCESS = 'SEQUENTIAL', STATUS = 'NEW',ERR=110)
        GOTO 120
110     CALL CLEARSCREEN( $GCLEARSCREEN )
        PRINT 1
        PRINT 1
        WRITE (*,70)
        PRINT 1
        WRITE (*,71)
        READ (*,'(A)') CHOICE
          IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
            CALL CLEARSCREEN( $GCLEARSCREEN )
            PRINT 22
            PRINT 1
            WRITE (*,72) FLNM
            OPEN (UNIT=0, FILE = FLNM(:LEN_TRIM(FLNM))//'.CST',
     +            ACCESS = 'SEQUENTIAL', STATUS = 'UNKNOWN')
          ELSE
            CALL CLEARSCREEN( $GCLEARSCREEN )
```

```
         CALL FILMGT
       END IF
120    OPEN (UNIT=5, FILE = FLNM(:LEN_TRIM(FLNM))//'.COR',
     +       ACCESS = 'SEQUENTIAL', STATUS = 'UNKNOWN')
       PRINT 12
       WRITE (*, '(1x,a\)') 'Enter number of cost elements: '
       READ (*,*,ERR=900) COUNT
C      Initialize Totals
       TCMEAN = 0
       TCVAR  = 0
       TCOST  = 0
       CORSUM = 0
       SUM    = 0
C      Begin Cost Data Input
         DO 160 I = 1,COUNT
140        CALL CLEARSCREEN( $GCLEARSCREEN )
           PRINT 12
           WRITE (*,2)
           WRITE (*,3)
           PRINT 17
           WRITE (*,4) I
           READ (*,'(A)') NAME(I)
           WRITE (*,5)
           READ (*,*,ERR=140) COST(I)
           WRITE (*,6)
           WRITE (*,7)
           READ (*,'(A)') DISTR(I)
             IF (DISTR(I).EQ.'N'.OR.DISTR(I).EQ.'n'.OR.
     +           DISTR(I).EQ.'L'.OR.DISTR(I).EQ.'l') THEN
               WRITE(*,77)
               READ (*,*,ERR=140) SELECT
               IF (SELECT.EQ.1) GOTO 145
               WRITE (*,78)
               READ (*,*,ERR=140) STDDEV(I)
               GOTO 147
             ELSE
               STDDEV(I)=0
             END IF
145        PRINT 8
           READ (*,*,ERR=140) MIN(I)
           PRINT 10
           READ (*,*,ERR=140) MAX(I)
C      Check MINIMUM < Mode < MAXIMUM
             IF (COST(I).LT.MIN(I).AND.COST(I).GT.MAX(I)) THEN
               CALL CLEARSCREEN( $GCLEARSCREEN )
               PRINT 1
               PRINT 1
               WRITE (*,24)
               PRINT 1
               WRITE (*,25)
               WRITE (*,26)
               PRINT 1
               PRINT 1
               PRINT 13
               READ (*,*)
```

```
            GOTO 140
          ELSE IF (COST(I).GT.MAX(I)) THEN
            CALL CLEARSCREEN( $GCLEARSCREEN )
            PRINT 1
            PRINT 1
            WRITE (*,24)
            PRINT 1
            WRITE (*,25)
            PRINT 1
            PRINT 1
            PRINT 13
            READ (*,*)
            GOTO 140
          ELSE IF (COST(I).LT.MIN(I)) THEN
            CALL CLEARSCREEN( $GCLEARSCREEN )
            PRINT 1
            PRINT 1
            WRITE (*,24)
            PRINT 1
            WRITE (*,26)
            PRINT 1
            PRINT 1
            PRINT 13
            READ (*,*)
            GOTO 140
          END IF
147     PRINT 11
        READ (*,'(A)') CHOICE
          IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
            CORR(I) = 1
          ELSE
            CORR(I) = 0
          ENDIF
        IF (DISTR(I) .EQ. 'U' .OR. DISTR(I) .EQ. 'u') THEN
C     Uniform Distribution
          CMEAN(I)=(MIN(I)+MAX(I))/2
          CVAR(I)=((MAX(I)-MIN(I))**2)/12
        ELSE IF (DISTR(I) .EQ. 'T' .OR. DISTR(I) .EQ. 't') THEN
C     Triangular Distribution
          CMEAN(I)=(MIN(I)+COST(I)+MAX(I))/3
          CVAR(I)=(MIN(I)*(MIN(I)-COST(I))+
     +          MAX(I)*(MAX(I)-MIN(I))+
     +          COST(I)*(COST(I)-MAX(I)))/18
        ELSE IF (DISTR(I) .EQ. 'N' .OR. DISTR(I) .EQ. 'n') THEN
C     Normal Distribution
          IF (SELECT.EQ.2) THEN
            CVAR(I)=STDDEV(I)**2
            MAX(I)=COST(I)+3*STDDEV(I)
            MIN(I)=COST(I)-3*STDDEV(I)
          ELSE
C     Insure Symmetry of endpoints
            IF (ABS(MAX(I)-COST(I)) .GT. ABS(COST(I)-MIN(I))) THEN
              MIN(I)=COST(I)-(MAX(I)-COST(I))
            ELSE
              MAX(I)=(COST(I)-MIN(I))+COST(I)
```

```
                END IF
              CVAR(I)=((MAX(I)-MIN(I))/6)**2
              END IF
            CMEAN(I)=COST(I)
            ELSE IF (DISTR(I) .EQ. 'L' ..OR. DISTR(I) .EQ. 'l') THEN
C     Lognormal Distribution
                IF (SELECT.EQ.2) THEN
                  LVAR=STDDEV(I)**2
                  LMEAN=LOG(COST(I))+LVAR
                  LMIN=LMEAN-(3*SQRT(LVAR))
                  LMAX=(3*SQRT(LVAR))+LMEAN
                  MIN(I)=EXP(LMIN)
                  MAX(I)=EXP(LMAX)
                  GOTO 153
                ELSE
                  LMAX=LOG(MAX(I))
                    IF (MIN(I).LE.0) THEN
                      LMIN=0
                    ELSE
                      LMIN=LOG(MIN(I))
                    END IF
                  LVAR=((LMAX-LMIN)/6)**2
                  LMEAN=((LMAX-LMIN)/2)+LMIN
                END IF
153           CMEAN(I)=EXP(LMEAN+(.5*LVAR))
              CVAR(I)=EXP((2*LMEAN)+LVAR)*(EXP(LVAR)-1)
            ELSE IF (DISTR(I) .EQ. 'B' .OR. DISTR(I) .EQ. 'b') THEN
C     BETA Distribution
              CMEAN(I)=(MIN(I)+4*COST(I)+MAX(I))/6
              CVAR(I)=((MAX(I)-MIN(I))**2)/36
            ELSE
              GOTO 140
            END IF
155         CORSUM = CORSUM + CORR(I)
            WRITE (*,38)
            READ (*,'(A)') CHOICE
            IF (CHOICE.EQ.'N' .OR. CHOICE.EQ.'n') GOTO 140
            PRINT 1
            PRINT 20
            PRINT 21,NAME(I),MIN(I),COST(I),MAX(I),
     +             DISTR(I),CMEAN(I),CVAR(I)
            PRINT 13
            READ (*,*)
C     Write cost data to file ('FLNM'.CST)
            WRITE (0,23) I,NAME(I),MIN(I),COST(I),MAX(I),DISTR(I),
     +                   CORR(I),STDDEV(I)
160       CONTINUE
C     Build correlation matrix template
495       DO 170 L=1,COUNT
            DO 180 M=1,COUNT
              IF (L.EQ.M) THEN
                CORMAT(L,M)=1
              ELSE
                CORMAT(L,M)=0
              END IF
```

```
180         CONTINUE
170       CONTINUE
          IF (CORSUM.EQ.0) GOTO 700
C       Formulate Correlation Matrix
          DO 520 J=1,COUNT
            IF (CORR(J).EQ.1) THEN
530           CALL CLEARSCREEN( $GCLEARSCREEN )
              WRITE (*,30)
              PRINT 12
              WRITE (*,31) J,NAME(J)
              PRINT 12
              WRITE (*,79)
              WRITE (*,80)
                DO 532 K=1,COUNT
                  IF (CORMAT(K,J).NE.0 .AND. K.NE.J) THEN
                    WRITE (*,81) K,NAME(K),CORMAT(K,J)
                  END IF
                  IF (CORMAT(J,K).NE.0 .AND. J.NE.K) THEN
                    WRITE (*,81) K,NAME(K),CORMAT(J,K)
                  END IF
532             CONTINUE
540           WRITE (*,39)
              READ (*,'(A)') CHOICE
              IF (CHOICE.EQ.'N' .OR. CHOICE.EQ.'n') GOTO 520
              WRITE (*,32)
              READ (*,*,ERR=530) WITH
              IF (WITH.GT.COUNT .OR. J.EQ.WITH) GOTO 530
              WRITE (*,33)
              READ (*,*,ERR=530) RHO(J)
              IF (RHO(J).LT.-1 .OR. RHO(J).GT.1) GOTO 530
                IF (CORMAT(WITH,J).NE.RHO(J).AND.CORMAT(WITH,J).NE.0) THEN
                  CALL CLEARSCREEN( $GCLEARSCREEN )
                  PRINT 1
                  PRINT 1
                  WRITE (*,73)
                  PRINT 12
                  WRITE (*,34)
                  WRITE (*,35)
                  WRITE (*,36)
                  WRITE (*,37) J,NAME(J),WITH,NAME(WITH),CORMAT(WITH,J)
                  PRINT 12
                  WRITE (*,74)
                  READ (*,'(A)') CHOICE
                  PRINT 12
                    IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
                      CALL CLEARSCREEN( $GCLEARSCREEN )
                      CORMAT(WITH,J)=RHO(J)
                      WRITE (*,30)
                      PRINT 12
                      WRITE (*,31) J,NAME(J)
                      PRINT 12
                      WRITE (*,75) WITH
                      PRINT 12
                      WRITE (*,76) RHO(J)
                      PRINT 1
```

```
                 GOTO 535
               ELSE
                 GOTO 530
               END IF
           ELSE
             PRINT 12
             CORMAT(J,WITH)=RHO(J)
           END IF
535      WRITE (*,34)
         WRITE (*,35)
         WRITE (*,36)
         WRITE (*,37) J,NAME(J),WITH,NAME(WITH),RHO(J)
         WRITE (*,38)
         READ (*,'(A)') CHOICE
         IF (CHOICE.EQ.'N' .OR. CHOICE.EQ.'n') THEN
           CORMAT(J,WITH)=0
           GOTO 530
         ELSE
           GOTO 530
         END IF
       ELSE
         GOTO 520
       END IF
520    CONTINUE
C     Check correlation matrix symmetry
       DO 620 L=1,COUNT
         DO 630 M=1,COUNT
           IF (L.EQ.M) THEN
             CORMAT(L,M)=1
           ELSE IF (CORMAT(L,M).EQ.CORMAT(M,L)) THEN
             GOTO 630
           ELSE IF (CORMAT(L,M).EQ.0 .AND. CORMAT(M,L).NE.0) THEN
             CORMAT(L,M)=CORMAT(M,L)
           ELSE IF (CORMAT(L,M).NE.0 .AND. CORMAT(M,L).EQ.0) THEN
             CORMAT(M,L)=CORMAT(L,M)
           END IF
630      CONTINUE
620    CONTINUE
C     Construct 'TEST' correlation matrix for JACOBI subroutine
       DO 640 L=1,COUNT
         DO 650 M=1,COUNT
           IF (L.EQ.M) THEN
             TEST(L,M)=1
           ELSE
             TEST(L,M)=CORMAT(L,M)
          END IF
650      CONTINUE
640    CONTINUE
C     Compute Eigenvalues
      CALL CLEARSCREEN( $GCLEARSCREEN )
      CALL jacobi (TEST,COUNT,NP,D,V,NROT)
       DO 670 J=1,NP
           IF (D(J).GE.0) THEN
             GOTO 670
           ELSE
```

```
                PRINT 12
                PRINT 12
                PRINT 12
                WRITE (*,40)
                PRINT 12
                WRITE (*,13)
                READ (*,*)
                GOTO 495
              END IF
670        CONTINUE
C     Write correlation values to file ('FLNM'.COR)
700     DO 710 L=1,COUNT
          DO 720 M=1,COUNT
            WRITE (5,60) CORMAT(L,M)
720       CONTINUE
710     CONTINUE
      IF (CORSUM.EQ.0) GOTO 900
C     Display correlation values
800   CALL CLEARSCREEN( $GCLEARSCREEN )
      LINE = 5
      WRITE (*,50)
      PRINT 12
      WRITE (*,51)
      WRITE (*,52)
      WRITE (*,53)
        DO 810 L=1,COUNT-1
          DO 820 K=L+1,COUNT
            IF (CORMAT(L,K).NE.0) THEN
              SUM=SUM+1
            END IF
820       CONTINUE
          IF (SUM.EQ.0) GOTO 810
          WRITE (*,54) L,NAME(L)
            DO 830 M=L+1,COUNT
              IF (CORMAT(L,M).EQ.0) GOTO 830
              WRITE (*,55) M,NAME(M),CORMAT(L,M)
              LINE=LINE+1
              IF (LINE.EQ.21) THEN
                WRITE (*,13)
                CALL CLEARSCREEN( $GCLEARSCREEN )
                WRITE (*,57)
                PRINT 12
                WRITE (*,51)
                WRITE (*,52)
                WRITE (*,53)
                LINE=0
              END IF
830         CONTINUE
          WRITE (*,56)
          LINE=LINE+2
          SUM = 0
810     CONTINUE
      PRINT 13
      READ (*,*)
1     FORMAT ('0')
```

```
2       FORMAT ('0',31X,'COST UNCERTAINTY')
3       FORMAT (' ',34X,'Data Input')
4       FORMAT (' ',6X,I3,14X,'CES Name                     ',\)
5       FORMAT ('0',23X,'Estimated Cost (mode)     ',\)
6       FORMAT ('0',23X,'Distribution Shape')
7       FORMAT (' ',23X,'(U,T,N,L,B)              ',\)
8       FORMAT ('0',33X,'Minimum         ',\)
10      FORMAT (' ',33X,'Maximum         ',\)
11      FORMAT (' ',23X,'Correlated? (Y/N)         ',\)
12      FORMAT (' ')
13      FORMAT ('0',26X,'Press [RETURN] to Continue',\)
14      FORMAT (' ',21X,'Number of elements?                ',\)
15      FORMAT (' ',16X,'Correlated with element?     ',\)
16      FORMAT (' ',29X,'Coefficient     ',\)
17      FORMAT (' ',3X,'Element #')
20      FORMAT ('+','NAME',18X,'MIN',8X,'COST',9X,'MAX',2X,
     +          'DISTR',6X,'MEAN',9X,'VAR')
21      FORMAT (' ',A10,3X,3F12.1,4X,A1,2F12.1)
22      FORMAT ('0'31X,'CREATE DATA FILE')
23      FORMAT (I3,A10,3F15.4,A1,I1,F8.5)
24      FORMAT ('0'29X,'Check Input Values')
25      FORMAT ('0'22X,'Maximum value less than cost mode')
26      FORMAT ('0'20X,'Minimum value greater than cost mode')
30      FORMAT ('0'26X,'CREATE CORRELATION MATRIX')
31      FORMAT (' ',10X,'Cost Element Number:',1X,I3,12X,
     +                   'Cell Name:',1X,A10)
32      FORMAT ('0',24X,'Correlated With Element:     ',\)
33      FORMAT ('0',24X,'Correlation Coefficient:  ',\)
34      FORMAT (' ',4X,'Cost',9X,'Cell',7X,'Correlated With',
     +               6X,'Cell',8X,'Correlation')
35      FORMAT (' ',3X,'Element',7X,'Name'8X,'Cost Element',
     +              8X,'Name',8X,'Coefficient')
36      FORMAT (' ',3X,'-------',6X,'------'7X,'------------',
     +              7X,'------',7X,'-----------')
37      FORMAT (' ',5X,I3,6X,A10,9X,I3,10X,A10,8X,F5.2)
38      FORMAT ('0',27X,'Is this correct? (Y/N) ',\)
39      FORMAT (' ',29X,'Add correlation? (Y/N) ',\)
40      FORMAT ('0',21X,'Correlation Matrix is Inconsistent')
50      FORMAT ('0',30X,'CORRELATION VALUES')
51      FORMAT (' ',13X,'Element',5X,'Cell')
52      FORMAT (' ',13X,'Number',6X,'Name')
53      FORMAT (' ',13X,'------',6X,'----')
54      FORMAT (' ',15X,I3,7X,A10,6X,'correlated with:')
55      FORMAT (' ',39X,I3,2X,A10,2X,F5.2)
56      FORMAT (' ',13X,'-----------------------------------------------')
57      FORMAT ('0',30X,'Correlation Values')
60      FORMAT (F5.2)
70      FORMAT (' ',28X,'File already exists')
71      FORMAT ('0',21X,'Do you want to overwrite it? (Y/N) ',\)
72      FORMAT (' ','Enter Filename:',1X,A8)
73      FORMAT (' ',34X,'Currently')
74      FORMAT (' ',32X,'Replace? (Y/N) ',\)
75      FORMAT ('0',24X,'Correlated With Element:',3X,I3)
76      FORMAT (' ',24X,'Correlation Coefficient:',1X,F5.2)
77      FORMAT ('0',15X,'Select input type  1) Min/Max  2) Std Dev  ',\)
```

```
78      FORMAT ('0',23X,'Standard Deviation         ',\)
79      FORMAT (' ',5X,'Current Correlations')
80      FORMAT (' ',5X,'--------------------')
81      FORMAT (' ',4X,I3,2X,A10,2X,F5.2)
900     CLOSE (0, STATUS='KEEP')
        CLOSE (5, STATUS='KEEP')
        CALL CLEARSCREEN( $GCLEARSCREEN )
        CALL FILMGT
        END
C
C       Edit Cost and Correlation Data File
C
        SUBROUTINE EDIT
        REAL MIN,COST,MAX,MINTMP,CSTTMP,MAXTMP,CORMAT,TEMP,TEST,RHO
        DOUBLE PRECISION LMIN,LMEAN,LMAX,LVAR,STDDEV,STDTMP
        PARAMETER (NP=200)
        INTEGER COUNT, I, C, CORR, DEL, WITH, CORSUM, NROT
        CHARACTER FLNM*8, DISTR*1, NAME*10, NAMTMP*10
        DIMENSION COUNT(NP),NAME(NP),MIN(NP),COST(NP),MAX(NP),D(NP),
     +            V(NP,NP),CORMAT(NP,NP),TEMP(NP,NP),TEST(NP,NP),
     +            CORR(NP),DISTR(NP),STDDEV(NP)
        CALL CLEARSCREEN( $GCLEARSCREEN )
        I      = 0
        SUM    = 0
        LINE   = 0
        CORSUM = 0
        WRITE (*,10)
        PRINT 2
        WRITE (*, '(1x,a\)') 'Enter filename:  '
        READ (*,'(A)')FLNM
        OPEN (UNIT=3,FILE=FLNM(:LEN_TRIM(FLNM))//'.CST',
     +      STATUS='OLD',ACCESS='SEQUENTIAL',ERR=100)
        GOTO 105
100     CALL CLEARSCREEN( $GCLEARSCREEN )
        PRINT 1
        PRINT 2
        PRINT 2
        WRITE (*,4)
        PRINT 1
        PRINT 2
        PRINT 99
        READ (*,*)
        CALL CLEARSCREEN( $GCLEARSCREEN )
        CALL FILMGT
105       DO WHILE (.NOT. EOF(3))
            I=I+1
            READ (3,30) COUNT(I),NAME(I),MIN(I),COST(I),MAX(I),
     +                  DISTR(I),CORR(I),STDDEV(I)
            CORSUM=CORSUM+CORR(I)
          END DO
        CLOSE (3)
C       Read Correlation File
        OPEN (UNIT=5, FILE = FLNM(:LEN_TRIM(FLNM))//'.COR',
     +      ACCESS = 'SEQUENTIAL', STATUS = 'UNKNOWN',ERR=100)
          DO 110 L=1,I
```

```
          DO 115 M=1,I
            READ (5,5) CORMAT(L,M)
            TEMP(L,M)=CORMAT(L,M)
115       CONTINUE
110     CONTINUE
      CLOSE (5)
140   CALL CLEARSCREEN( $GCLEARSCREEN )
      PRINT 2
      PRINT 2
      WRITE (*,10)
      PRINT 2
      WRITE (*,11)
      WRITE (*,12)
      WRITE (*,8)
      PRINT 1
      WRITE (*,9)
      READ (*,*,ERR=140) SELECT
      IF (SELECT.GT.3 .OR. SELECT.LT.1) GOTO 140
C     Edit Cost Data File
        IF (SELECT.EQ.1) THEN
150       CALL CLEARSCREEN( $GCLEARSCREEN )
          WRITE (*,18) FLNM(:LEN_TRIM(FLNM))
          PRINT 1
          WRITE (*,25)
          WRITE (*,26)
            DO 600 C=1,I
              LINE=LINE+1
              WRITE (*,35) C,NAME(C),MIN(C),COST(C),MAX(C),
     +                     DISTR(C)
              IF (LINE.EQ.10.OR.C.EQ.I) THEN
                LINE = 0
                WRITE (*,13)
                READ (*,*,ERR=150) SELECT
C     Add Cost Element
                IF (SELECT.EQ.1) THEN
                  PRINT 1
                  WRITE (*,27)
                  READ (*,*,ERR=150) INS
                  IF (INS.LT.1) GOTO 150
                  I=I+1
                  INS=INS+1
                    IF (INS.GT.I) THEN
                      INS=I
                    END IF
                    DO 210 K=INS,I
                      NAMTMP = NAME(INS)
                      MINTMP = MIN(INS)
                      CSTTMP = COST(INS)
                      MAXTMP = MAX(INS)
                      DISTMP = DISTR(INS)
                      CORTMP = CORR(INS)
                      STDTMP = STDDEV(INS)
                      NAME(INS)    = NAME(K+1)
                      MIN(INS)     = MIN(K+1)
                      COST(INS)    = COST(K+1)
```

```
                MAX(INS)      = MAX(K+1)
                DISTR(INS)    = DISTR(K+1)
                CORR(INS)     = CORR(K+1)
                STDDEV(INS)   = STDDEV(K+1)
                NAME(K+1)     = NAMTMP
                MIN(K+1)      = MINTMP
                COST(K+1)     = CSTTMP
                MAX(K+1)      = MAXTMP
                DISTR(K+1)    = DISTMP
                CORR(K+1)     = CORTMP
                STDDEV(K+1)   = STDTMP
210           CONTINUE
215         CALL CLEARSCREEN( $GCLEARSCREEN )
            PRINT 1
            WRITE (*,40)
            WRITE (*,49)
            WRITE (*,41) INS
            READ (*,'(A)') NAME(INS)
            WRITE (*,43)
            READ (*,*,ERR=215) COST(INS)
            WRITE (*,45)
            WRITE (*,46)
            READ (*,'(A)') DISTR(INS)
              IF (DISTR(INS).EQ.'N'.OR.DISTR(INS).EQ.'n'.OR.
     +           DISTR(INS).EQ.'L'.OR.DISTR(INS).EQ.'l') THEN
                WRITE(*,36)
                READ (*,*,ERR=215) SELECT
                IF (SELECT.EQ.1) GOTO 220
                WRITE (*,37)
                READ (*,*,ERR=215) STDDEV(INS)
                GOTO 225
              ELSE
                 STDDEV(INS)=0
              END IF
220         WRITE (*,42)
            READ (*,*,ERR=215) MIN(INS)
            WRITE (*,44)
            READ (*,*,ERR=215) MAX(INS)
C     Check MINIMUM < Mode < MAXIMUM
              IF (COST(INS).LT.MIN(INS).AND.COST(INS).GT.
     +        MAX(INS)) THEN
                CALL CLEARSCREEN( $GCLEARSCREEN )
                PRINT 2
                PRINT 2
                WRITE (*,81)
                PRINT 2
                WRITE (*,82)
                WRITE (*,83)
                PRINT 2
                PRINT 2
                WRITE (*,99)
                READ (*,*)
                GOTO 215
              ELSE IF (COST(INS).GT.MAX(INS)) THEN
                CALL CLEARSCREEN( $GCLEARSCREEN )
```

```
                         PRINT 2
                         PRINT 2
                         WRITE (*,81)
                         PRINT 2
                         WRITE (*,82)
                         PRINT 2
                         PRINT 2
                         WRITE (*,99)
                         READ (*,*)
                         GOTO 215
                       ELSE IF (COST(INS).LT.MIN(INS)) THEN
                         CALL CLEARSCREEN( $GCLEARSCREEN )
                         PRINT 2
                         PRINT 2
                         WRITE (*,81)
                         PRINT 2
                         WRITE (*,83)
                         PRINT 2
                         PRINT 2
                         WRITE (*,99)
                         READ (*,*)
                         GOTO 215
                       END IF
225                  WRITE (*,47)
                     READ (*,'(A)') CHOICE
                       IF (CHOICE.EQ.'Y'.OR.CHOICE.EQ.'y') THEN
                         CORR(INS)=1
                       ELSE
                         CORR(INS)=0
                       END IF
                     WRITE (*,62)
                     READ (*,'(A)') CHOICE
                     IF (CHOICE.EQ.'N'.OR.CHOICE.EQ.'n') GOTO 215
                       IF (DISTR(INS).EQ.'U'.OR.DISTR(INS).EQ.'u') THEN
C     Uniform Distribution
                         CMEAN=(MIN(INS)+MAX(INS))/2
                         CVAR=((MAX(INS)-MIN(INS))**2)/12
                       ELSE IF (DISTR(INS).EQ.'T'.OR.DISTR(INS).EQ.'t')
     +                 THEN
C     Triangular Distribution
                         CMEAN=(MIN(INS)+COST(INS)+MAX(INS))/3
                         CVAR=(MIN(INS)*(MIN(INS)-COST(INS))+
     +                         MAX(INS)*(MAX(INS)-MIN(INS))+
     +                         COST(INS)*(COST(INS)-MAX(INS)))/18
                       ELSE IF (DISTR(INS).EQ.'N'.OR.DISTR(INS).EQ.'n')
     +                 THEN
C     Normal Distribution
                         IF (SELECT.EQ.2) THEN
                           CVAR=STDDEV(INS)**2
                           MAX(INS)=3*STDDEV(INS)+COST(INS)
                           MIN(INS)=COST(INS)-3*STDDEV(INS)
                         ELSE
C     Insure Symmetry of endpoints
                           IF (ABS(MAX(INS)-COST(INS)).GT.
     +                     ABS(COST(INS)-MIN(INS))) THEN
```

```
                      MIN(INS)=COST(INS)-(MAX(INS)-COST(INS))
                    ELSE
                      MAX(INS)=(COST(INS)-MIN(INS))+COST(INS)
                    END IF
                    CVAR=((MAX(INS)-MIN(INS))/6)**2
                  END IF
                  CMEAN=COST(INS)
                ELSE IF (DISTR(INS).EQ.'L'.OR.DISTR(INS).EQ.'l')
     +          THEN
C     Lognormal Distribution
                  IF (SELECT.EQ.2) THEN
                    LVAR=STDDEV(INS)**2
                    LMEAN=LOG(COST(INS))+LVAR
                    LMIN=LMEAN-(3*SQRT(LVAR))
                    LMAX=(3*SQRT(LVAR))+LMEAN
                    MIN(INS)=EXP(LMIN)
                    MAX(INS)=EXP(LMAX)
                  ELSE
                    LMAX=LOG(MAX(I))
                      IF (MIN(I).LE.0) THEN
                        LMIN=0
                      ELSE
                        LMIN=LOG(MIN(I))
                      END IF
                    LVAR=((LMAX-LMIN)/6)**2
                    LMEAN=((LMAX-LMIN)/2)+LMIN
                  END IF
                  CMEAN=EXP(LMEAN+(.5*LVAR))
                  CVAR=EXP((2*LMEAN)+LVAR)*(EXP(LVAR)-1)
                ELSE
C     BETA Distribution
                  CMEAN=(MIN(INS)+4*COST(INS)+MAX(INS))/6
                  CVAR=((MAX(INS)-MIN(INS))**2)/36
                END IF
              PRINT 2
              WRITE (*,38)
              WRITE (*,39) NAME(INS),MIN(INS),COST(INS),
     +                     MAX(INS),DISTR(INS),CMEAN,CVAR
              WRITE (*,99)
              READ (*,*)
C     Add variable to correlation matrix
                DO 230 L=1,I
                  DO 235 M=1,I
                    IF (L.EQ.M) THEN
                      TEMP(L,M)=1
                    ELSE IF (L.EQ.INS .OR. M.EQ.INS) THEN
                      TEMP(L,M)=0
                    ELSE IF (L.GT.INS) THEN
                      IF (M.GT.INS) THEN
                        TEMP(L,M)=TEMP(L-1,M-1)
                      ELSE
                        TEMP(L,M)=TEMP(L-1,M)
                      END IF
                    ELSE IF (M.GT.INS) THEN
                        TEMP(L,M)=TEMP(L,M-1)
```

```
                    ELSE
                      TEMP(L,M)=TEMP(L,M)
                    END IF
235               CONTINUE
230             CONTINUE
              IF (CORR(INS).EQ.0) GOTO 250
240           CALL CLEARSCREEN( $GCLEARSCREEN )
              WRITE (*,65)
              PRINT 1
              WRITE (*,66) INS,NAME(INS)
              PRINT 1
              WRITE (*,67)
              WRITE (*,68)
                DO 245 K=1,I
                  IF (TEMP(INS,K).NE.0 .AND. INS.NE.K) THEN
                    WRITE (*,69) K,NAME(K),TEMP(INS,K)
                  END IF
245             CONTINUE
              WRITE (*,70)
              READ (*,'(A)') CHOICE
              IF (CHOICE.EQ.'N' .OR. CHOICE.EQ.'n') GOTO 250
              WRITE (*,71)
              READ (*,*,ERR=240) WITH
              IF (WITH.GT.I .OR. WITH.EQ.INS) GOTO 240
              PRINT 1
              WRITE (*,72)
              READ (*,*,ERR=240) RHO
              IF (RHO.LT.-1 .OR. RHO.GT.1) GOTO 240
              TEMP (INS,WITH)=RHO
              TEMP (WITH,INS)=RHO
              PRINT 1
              WRITE (*,73)
              WRITE (*,74)
              WRITE (*,75)
              WRITE (*,76) INS,NAME(INS),WITH,NAME(WITH),RHO
              WRITE (*,77)
              READ (*,'(A)') CHOICE
                IF (CHOICE.EQ.'N' .OR. CHOICE.EQ.'n') THEN
                  TEMP(INS,WITH)=0
                  TEMP(WITH,INS)=0
                  GOTO 240
                ELSE
                  GOTO 240
                END IF
C     Construct 'TEST' correlation matrix for JACOBI subroutine
250             DO 255 L=1,I
                  DO 260 M=1,I
                    IF (L.EQ.M) THEN
                      TEST(L,M)=1
                    ELSE
                      TEST(L,M)=TEMP(L,M)
                      TEST(M,L)=TEMP(L,M)
                    END IF
260               CONTINUE
255             CONTINUE
```

```
                  GOTO 150
C     Delete Cost Element
                ELSE IF (SELECT.EQ.2) THEN
                  PRINT 1
                  WRITE (*,28)
                  READ (*,*,ERR=150) DEL
                  IF (DEL.LT.1 .OR. DEL.GT.I) GOTO 150
                  WRITE (*,21) DEL, NAME(DEL)
                  WRITE (*,22)
                  READ (*,'(A)') CHOICE
                    IF (CHOICE.EQ.'Y'.OR.CHOICE.EQ.'y') THEN
                      DO 300 K=DEL,I
                        COUNT(K)   = K
                        NAME(K)    = NAME(K+1)
                        MIN(K)     = MIN(K+1)
                        COST(K)    = COST(K+1)
                        MAX(K)     = MAX(K+1)
                        DISTR(K)   = DISTR(K+1)
                        CORR(K)    = CORR(K+1)
                        STDDEV(K)  = STDDEV(K+1)
300                   CONTINUE
                    ELSE
                      GOTO 150
                    END IF
C     Delete variable from correlation matrix
                    DO 305 L=1,I
                      DO 310 M=L+1,I
                        IF (L.EQ.I .OR. M.EQ.I) THEN
                          TEMP(L,M)=0
                        ELSE IF (L.EQ.M) THEN
                          TEMP(L,M)=1
                        ELSE IF (L.GE.DEL) THEN
                          TEMP(L,M)=TEMP(L+1,M+1)
                        ELSE IF (M.GE.DEL) THEN
                          TEMP(L,M)=TEMP(L,M+1)
                        END IF
310                   CONTINUE
305                 CONTINUE
C     Check correlation matrix symmetry
                    DO 315 L=1,I
                      DO 320 M=1,I
                        IF (L.EQ.M) THEN
                          TEMP(L,M)=1
                        ELSE IF (TEMP(L,M).EQ.TEMP(M,L)) THEN
                          GOTO 320
                        ELSE IF (TEMP(L,M).EQ.0.AND.TEMP(M,L).NE.0) THEN
                          TEMP(L,M)=TEMP(M,L)
                        ELSE IF (TEMP(L,M).NE.0.AND.TEMP(M,L).EQ.0) THEN
                          TEMP(M,L)=TEMP(L,M)
                        END IF
320                   CONTINUE
315                 CONTINUE
                  TEMP(I,I)=0
                  I=I-1
C     Construct 'TEST' correlation matrix for JACOBI subroutine
```

```
325                  DO 330 L=1,I
                       DO 335 M=1,I
                         IF (L.EQ.M) THEN
                           TEST(L,M)=1
                         ELSE
                           TEST(L,M)=TEMP(L,M)
                           TEST(M,L)=TEMP(L,M)
                         END IF
335                    CONTINUE
330                  CONTINUE
                   GOTO 150
C     Modify Cost Element
                 ELSE IF (SELECT.EQ.3) THEN
                   PRINT 1
                   WRITE (*,29)
                   READ (*,*,ERR=150) MOD
                   IF (MOD.LT.1 .OR. MOD.GT.I) GOTO 150
400                CALL CLEARSCREEN( $GCLEARSCREEN )
                   WRITE (*,48)
                   WRITE (*,49)
                   WRITE (*,50) MOD,NAME(MOD)
                   READ (*,'(A)') CHOICE
                     IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
                       WRITE (*,51)
                       READ (*,'(A)') NAMTMP
                       NAME(MOD)=NAMTMP
                     ELSE
                       NAMTMP=NAME(MOD)
                     END IF
                   WRITE (*,54) COST(MOD)
                   READ (*,'(A)') CHOICE
                     IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
                       WRITE (*,55)
                       READ (*,*,ERR=400) CSTTMP
                       COST(MOD)=CSTTMP
                     ELSE
                       CSTTMP=COST(MOD)
                     END IF
                   WRITE (*,58) DISTR(MOD)
                   READ (*,'(A)') CHOICE
                     IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
                       WRITE (*,59)
                       READ (*,'(A)') DISTMP
                         IF (DISTMP.EQ.'N'.OR.DISTMP.EQ.'n') THEN
                           DISTR(MOD)=DISTMP
                           WRITE(*,36)
                           READ (*,*,ERR=400) SELECT
                           IF (SELECT.EQ.1) GOTO 405
                           WRITE (*,34)
                           READ (*,*,ERR=400) STDDEV(MOD)
                           MAX(MOD)=3*STDDEV(MOD)+COST(MOD)
                           MIN(MOD)=COST(MOD)-3*STDDEV(MOD)
                           GOTO 410
                         ELSE IF (DISTMP.EQ.'L'.OR.DISTMP.EQ.'l') THEN
                           DISTR(MOD)=DISTMP
```

```
              WRITE(*,36)
              READ (*,*,ERR=400) SELECT
              IF (SELECT.EQ.1) GOTO 405
              WRITE (*,34)
              READ (*,*,ERR=400) STDDEV(MOD)
              LVAR=STDDEV(MOD)**2
              LMEAN=LOG(COST(MOD))+LVAR
              LMIN=LMEAN-(3*SQRT(LVAR))
              LMAX=(3*SQRT(LVAR))+LMEAN
              MIN(MOD)=EXP(LMIN)
              MAX(MOD)=EXP(LMAX)
              GOTO 410
            END IF
          DISTR(MOD)=DISTMP
        ELSE
          DISTMP=DISTR(MOD)
            IF (STDDEV(MOD).NE.0) THEN
              WRITE (*,32) STDDEV(MOD)
              READ (*,'(A)') CHOICE
                IF (CHOICE.EQ.'Y'.OR.CHOICE.EQ.'y') THEN
                  WRITE (*,33)
                  READ (*,*,ERR=400) STDTMP
                  STDDEV(MOD)=STDTMP
                    IF (DISTMP.EQ.'N'.OR.DISTMP.EQ.'n') THEN
                      MAX(MOD)=3*STDDEV(MOD)+COST(MOD)
                      MIN(MOD)=COST(MOD)-3*STDDEV(MOD)
                    ELSE IF (DISTMP.EQ.'L'.OR.
     +              DISTMP.EQ.'l') THEN
                      LVAR=STDDEV(MOD)**2
                      LMEAN=LOG(COST(MOD))+LVAR
                      LMIN=LMEAN-(3*SQRT(LVAR))
                      LMAX=(3*SQRT(LVAR))+LMEAN
                      MIN(MOD)=EXP(LMIN)
                      MAX(MOD)=EXP(LMAX)
                    END IF
                ELSE
                  STDTMP=STDDEV(MOD)
                END IF
              GOTO 412
            END IF
        END IF
405   STDDEV(MOD)=0
      WRITE (*,52) MIN(MOD)
      READ (*,'(A)') CHOICE
        IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
          WRITE (*,53)
          READ (*,*,ERR=400) MINTMP
          MIN(MOD)=MINTMP
        ELSE
          MINTMP=MIN(MOD)
        END IF
      WRITE (*,56) MAX(MOD)
      READ (*,'(A)') CHOICE
        IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
          WRITE (*,57)
```

```
                   READ (*,*,ERR=400) MAXTMP
                   MAX(MOD)=MAXTMP
                 ELSE
                   MAXTMP=MAX(MOD)
                 END IF
C     Check MINIMUM < Mode < MAXIMUM
410              IF (COST(MOD).LT.MIN(MOD).AND.COST(MOD).GT.
     +           MAX(MOD)) THEN
                   CALL CLEARSCREEN( $GCLEARSCREEN )
                   PRINT 2
                   PRINT 2
                   WRITE (*,81)
                   PRINT 2
                   WRITE (*,82)
                   WRITE (*,83)
                   PRINT 2
                   PRINT 2
                   WRITE (*,99)
                   READ (*,*)
                   GOTO 400
                 ELSE IF (COST(MOD).GT.MAX(MOD)) THEN
                   CALL CLEARSCREEN( $GCLEARSCREEN )
                   PRINT 2
                   PRINT 2
                   WRITE (*,81)
                   PRINT 2
                   WRITE (*,82)
                   PRINT 2
                   PRINT 2
                   WRITE (*,99)
                   READ (*,*)
                   GOTO 400
                 ELSE IF (COST(MOD).LT.MIN(MOD)) THEN
                   CALL CLEARSCREEN( $GCLEARSCREEN )
                   PRINT 2
                   PRINT 2
                   WRITE (*,81)
                   PRINT 2
                   WRITE (*,83)
                   PRINT 2
                   PRINT 2
                   WRITE (*,99)
                   READ (*,*)
                   GOTO 400
                 END IF
                   IF (SELECT.EQ.1) THEN
                     IF (DISTR(MOD).EQ.'N'.OR.DISTR(MOD).EQ.'n') THEN
C     Insure Symmetry of endpoints
                       IF (ABS(MAX(MOD)-COST(MOD)) .GT.
     +                 ABS(COST(MOD)-MIN(MOD))) THEN
                         MIN(MOD)=COST(MOD)-(MAX(MOD)-COST(MOD))
                       ELSE
                         MAX(MOD)=(COST(MOD)-MIN(MOD))+COST(MOD)
                       END IF
                     END IF
```

```
            END IF
412         IF (CORR(MOD).EQ.0) THEN
              WRITE (*,60) 'No'
            ELSE
              WRITE (*,60) 'Yes'
            END IF
          READ (*,'(A)') CHOICE
            IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
              WRITE (*,61)
              READ (*,'(A)') CHOICE
                IF (CHOICE.EQ.'Y'.OR.CHOICE.EQ.'y') THEN
                  CORTMP=1
                ELSE IF (CHOICE.EQ.'N'.OR.CHOICE.EQ.'n') THEN
                  CORTMP=0
                    DO 415 F=1,I
                      IF (F.EQ.MOD) THEN
                        TEMP(F,MOD)=1
                      ELSE
                        TEMP(F,MOD)=0
                        TEMP(MOD,F)=0
                      END IF
415                 CONTINUE
                END IF
              CORR(MOD)=CORTMP
            ELSE
              CORTMP=CORR(MOD)
            END IF
          WRITE (*,99)
          READ (*,*)
          IF (CORR(MOD).EQ.0) GOTO 150
420       CALL CLEARSCREEN( $GCLEARSCREEN )
          WRITE (*,65)
          PRINT 1
          WRITE (*,66) MOD,NAME(MOD)
          PRINT 1
          WRITE (*,67)
          WRITE (*,68)
            DO 425 K=1,I
              IF (TEMP(K,MOD).NE.0 .AND. K.NE.MOD) THEN
                WRITE (*,69) K,NAME(K),TEMP(K,MOD)
              END IF
425         CONTINUE
          WRITE (*,70)
          READ (*,'(A)') CHOICE
          IF (CHOICE.EQ.'N' .OR. CHOICE.EQ.'n') GOTO 430
          WRITE (*,71)
          READ (*,*,ERR=420) WITH
          IF (WITH.GT.I .OR. WITH.EQ.MOD) GOTO 420
          PRINT 1
          WRITE (*,72)
          READ (*,*,ERR=420) RHO
          IF (RHO.LT.-1 .OR. RHO.GT.1) GOTO 420
          TEMP (WITH,MOD)=RHO
          TEMP (MOD,WITH)=RHO
          PRINT 1
```

```
                  WRITE (*,73)
                  WRITE (*,74)
                  WRITE (*,75)
                  WRITE (*,76) MOD,NAME(MOD),WITH,NAME(WITH),RHO
                  WRITE (*,77)
                  READ (*,'(A)') CHOICE
                    IF (CHOICE.EQ.'N' .OR. CHOICE.EQ.'n') THEN
                      TEMP(MOD,WITH)=0
                      TEMP(WITH,MOD)=0
                      GOTO 420
                    ELSE
                      GOTO 420
                    END IF
C     Construct 'TEST' correlation matrix for JACOBI subroutine
430                   DO 435 L=1,I
                        DO 440 M=1,I
                          IF (L.EQ.M) THEN
                            TEST(L,M)=1
                          ELSE
                            TEST(L,M)=TEMP(L,M)
                            TEST(M,L)=TEMP(L,M)
                          END IF
440                     CONTINUE
435                   CONTINUE
                    GOTO 150
C     Return to 'EDIT' menu
              ELSE IF (SELECT.EQ.5) THEN
                      DO 500 L=1,I
                        DO 505 M=1,I
                          IF (L.EQ.M) THEN
                            TEST(L,M)=1
                          ELSE
                            TEST(L,M)=TEMP(L,M)
                            TEST(L,M)=TEMP(L,M)
                          END IF
505                     CONTINUE
500                   CONTINUE
C     Re-compute Eigenvalues
                    CALL CLEARSCREEN( $GCLEARSCREEN )
                    CALL jacobi (TEST,I,NP,D,V,NROT)
                      DO 510 L=1,I
                        DO 515 M=1,I
                          IF (L.EQ.M) THEN
                            TEMP(M,L)=1
                          ELSE
                            TEMP(L,M)=TEST(L,M)
                            TEMP(M,L)=TEST(L,M)
                          END IF
515                     CONTINUE
510                   CONTINUE
                      DO 520 J=1,I
                        IF (D(J).LT.0) THEN
                          FAIL=FAIL+1
                        END IF
520                   CONTINUE
```

```
                         IF (FAIL.GT.0) THEN
                           FAIL=0
                           PRINT 1
                           PRINT 2
                           WRITE (*,80)
                           PRINT 1
                           WRITE (*,99)
                           READ (*,*)
                           GOTO 140
                         END IF
                 GOTO 140
C     Continue to next page
               ELSE IF (SELECT.EQ.4) THEN
                 CALL CLEARSCREEN( $GCLEARSCREEN )
                 WRITE (*,18) FLNM(:LEN_TRIM(FLNM))
                 PRINT 1
                 WRITE (*,25)
                 WRITE (*,26)
                 GOTO 600
               ELSE
                 GOTO 150
               END IF
             END IF
600        CONTINUE
         GOTO 150
C     Save Cost Data File
       ELSE IF (SELECT.EQ.2) THEN
700      CALL CLEARSCREEN( $GCLEARSCREEN )
         PRINT 2
         WRITE (*,84)
         PRINT 2
         PRINT 1
         WRITE (*, '(1x,a\)') 'Save file as:  '
         READ (*,'(A)') FLNM
         OPEN (UNIT=4, FILE = FLNM(:LEN_TRIM(FLNM))//'.CST',
     +         ACCESS = 'SEQUENTIAL', STATUS = 'NEW',ERR=710)
         GOTO 720
710      CALL CLEARSCREEN( $GCLEARSCREEN )
         PRINT 2
         PRINT 2
         WRITE (*,87)
         PRINT 2
         WRITE (*,88)
         READ (*,'(A)') CHOICE
           IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
             CALL CLEARSCREEN( $GCLEARSCREEN )
             OPEN (UNIT=4, FILE = FLNM(:LEN_TRIM(FLNM))//'.CST',
     +             ACCESS = 'SEQUENTIAL', STATUS = 'UNKNOWN')
             GOTO 720
           ELSE
             GOTO 700
           END IF
720      OPEN (UNIT=6,FILE=FLNM(:LEN_TRIM(FLNM))//'.COR',
     +         STATUS='UNKNOWN',ACCESS='SEQUENTIAL',ERR=100)
           DO 800 K=1,I
```

```
              WRITE (4,63) K,NAME(K),MIN(K),COST(K),MAX(K),
     +                     DISTR(K),CORR(K),STDDEV(K)
 800        CONTINUE
          CLOSE(4)
C     Save Correlation Data File
            DO 805 L=1,I
              DO 810 M=1,I
                CORMAT(L,M)=TEMP(L,M)
                WRITE (6,5) CORMAT(L,M)
 810          CONTINUE
 805        CONTINUE
          CLOSE(6)
          CALL CLEARSCREEN( $GCLEARSCREEN )
          PRINT 2
          PRINT 2
          PRINT 2
          WRITE (*,85) FLNM(:LEN_TRIM(FLNM))
          WRITE (*,86) FLNM(:LEN_TRIM(FLNM))
          PRINT 2
          PRINT 2
          PRINT 2
          PRINT 99
          READ (*,*)
          GOTO 140
C     Call Subroutine 'FILMGT'
        ELSE IF (SELECT.EQ.3) THEN
          CALL CLEARSCREEN( $GCLEARSCREEN )
          CALL FILMGT
        END IF
1     FORMAT (' ')
2     FORMAT ('0')
3     FORMAT (F5.2)
4     FORMAT ('0',29X,'File Does Not Exist')
5     FORMAT (F5.2)
8     FORMAT (' ',26X,'3)   Return to File Management')
9     FORMAT ('0',31X,'Enter Selection:  ',\)
10    FORMAT ('0',31X,'EDIT DATA FILE')
11    FORMAT (' ',26X,'1)   Edit Cost Data')
12    FORMAT (' ',26X,'2)   Save File')
13    FORMAT ('0',3X,'1) Insert',4X,'2) Delete',4X,'3) Modify',4X,
     +           '4) Next Page',4X,'5) Return    ',\)
14    FORMAT (' ',26X,'2)   Add Cost Element')
15    FORMAT (' ',26X,'3)   Delete Cost Element')
16    FORMAT (' ',26X,'4)   Change Cost Element')
17    FORMAT (' ',26X,'5)   Return to Previous Menu')
18    FORMAT ('0',31X,'EDIT COST DATA'12X,A8,'.CST')
19    FORMAT ('0',31X,'EDIT CORRELATION VALUES')
20    FORMAT (' ',26X,'1)   Modify Cost Element')
21    FORMAT ('0','Delete CES number:  ',I3,2X,A10)
22    FORMAT (' ',4X,'Confirm (Y/N): ',\)
25    FORMAT (' ',5X,'NAME',16X,'MIN',8X,'MODE',9X,'MAX'
     +           ,5X,'Distribution')
26    FORMAT (' ',5X,'----',16X,'---',8X,'----',9X,'---'
     +           ,5X,'------------')
27    FORMAT ('0','Insert after element number:  ',\)
```

```
28    FORMAT ('0','Enter CES number to delete:  ',\)
29    FORMAT ('0','Enter CES number to modify:  ',\)
30    FORMAT (I3,A10,3F15.4,A1,I1,F8.5)
32    FORMAT ('0',10X,'Standard Deviation:     ',F8.5,
     +          '     Replace (Y/N)  ',\)
33    FORMAT (' ',21X,'Std Dev:  ',\)
34    FORMAT ('0',10X,'Standard Deviation:              ',\)
35    FORMAT (I3,3X,A10,1X,3F12.1,10X,A1)
36    FORMAT ('0',15X,'Select input type  1) Min/Max  2) Std Dev  ',\)
37    FORMAT ('0',23X,'Standard Deviation        ',\)
38    FORMAT ('+','NAME',18X,'MIN',8X,'COST',9X,'MAX',2X,
     +  'DISTR',6X,'MEAN',9X,'VAR')
39    FORMAT (' ',A10,3X,3F12.1,4X,A1,2F12.1)
40    FORMAT ('0',36X,'INSERT')
41    FORMAT (' ',6X,I3,14X,'CES Name                    ',\)
42    FORMAT ('0',33X,'Minimum          ',\)
43    FORMAT ('0',23X,'Estimated Cost (mode)     ',\)
44    FORMAT (' ',33X,'Maximum          ',\)
45    FORMAT ('0',23X,'Distribution Shape')
46    FORMAT (' ',23X,'(U,T,N,L,B)               ',\)
47    FORMAT (' ',23X,'Correlated? (Y/N)         ',\)
48    FORMAT ('0',35X,'REPLACE')
49    FORMAT ('0',3X,'Element #')
50    FORMAT (' ',6X,I3,11X,'CES Name: ',A10,'      Replace (Y/N)  ',\)
51    FORMAT (' ',24X,'Name: ',\)
52    FORMAT ('0',15X,'Minimum Value: ',F12.1,'    Replace (Y/N)  ',\)
53    FORMAT (' ',25X,'Min:  ',\)
54    FORMAT ('0',13X,'Estimate (mode): ',F12.1,'    Replace (Y/N)  ',\)
55    FORMAT (' ',24X,'Mode:  ',\)
56    FORMAT ('0',15X,'Maximum Value: ',F12.1,'    Replace (Y/N)  ',\)
57    FORMAT (' ',25X,'Max:  ',\)
58    FORMAT ('0',16X,'Distribution:  ',A11,'    Replace (Y/N)  ',\)
59    FORMAT (' ',11X,'Enter (U,T,N,L,B):  ',\)
60    FORMAT ('0',11X,'Correlated? (Y/N):  ',A11,
     +          '     Replace (Y/N)  ',\)
61    FORMAT (' ',17X,'Enter (Y/N):  ',\)
62    FORMAT ('0',27X,'Is this correct? (Y/N) ',\)
63    FORMAT (I3,A10,3F15.4,A1,I1,F8.5)
64    FORMAT ('0',26X,'File: ',A8,'.CST saved')
65    FORMAT ('0',28X,'EDIT CORRELATION MATRIX')
66    FORMAT (' ',10X,'Cost Element Number:',1X,I3,12X,
     +          'Cell Name:',1X,A10)
67    FORMAT (' ',5X,'Current Correlations')
68    FORMAT (' ',5X,'-------------------')
69    FORMAT (' ',4X,I3,2X,A10,2X,F5.2)
70    FORMAT (' ',29X,'Add/change values? (Y/N) ',\)
71    FORMAT ('0',24X,'Correlated With Element:    ',\)
72    FORMAT (' ',24X,'Correlation Coefficient:  ',\)
73    FORMAT (' ',4X,'Cost',9X,'Cell',7X,'Correlated With',
     +       6X,'Cell',8X,'Correlation')
74    FORMAT (' ',3X,'Element',7X,'Name'8X,'Cost Element',
     +      8X,'Name',8X,'Coefficient')
75    FORMAT ('-',3X,'-------',6X,'------'7X,'------------',
     +      7X,'------',7X,'-----------')
76    FORMAT (' ',5X,I3,6X,A10,9X,I3,10X,A10,8X,F5.2)
```

```
77     FORMAT ('0',27X,'Is this correct? (Y/N) ',\)
80     FORMAT ('0',21X,'Correlation Matrix is Inconsistent')
81     FORMAT ('0'29X,'Check Input Values')
82     FORMAT ('0'22X,'Maximum value less than cost mode')
83     FORMAT ('0'20X,'Minimum value greater than cost mode')
84     FORMAT ('0',34X,'SAVE FILE')
85     FORMAT ('0',26X,'Files: ',A8,'.CST saved')
86     FORMAT (' ',33X,A8,'.COR saved')
87     FORMAT (' ',28X,'File already exists')
88     FORMAT ('0',21X,'Do you want to overwrite it? (Y/N) ',\)
99     FORMAT ('0',26X,'Press [RETURN] to Continue',\)
       END
C
C      Cost Uncertainty Module
C
       SUBROUTINE CSTUNC
       PARAMETER (NP=200)
       DIMENSION COUNT(NP),NAME(NP),CORMAT(NP,NP),MIN(NP),COST(NP),
     +           MAX(NP),CORR(NP),STDDEV(NP),CVAR(NP),DISTR(NP)
       DIMENSION PROB(11),LPROB(11),ZSCORE(11),PCENT(11)
       DOUBLE PRECISION LMIN,LMEAN,LMAX,LVAR,TCLMEAN,TCLVAR,LPROB,LEST,
     +                  STDDEV,CMEAN,CVAR,AREA,NEWCST,X,Z
       REAL MIN,COST,MAX,TCMEAN,TCVAR,PROB,EST,ZSCORE
       INTEGER COUNT, K, L, M, CORR
       CHARACTER FLNM*8, DISTR*1, NAME*10, PCENT*3
       DATA ZSCORE /-1.28155,-.84162,-.5244,-.25335,0,.25335,
     +              .5244,.84162,1.28155,1.64485,2.32635/
       DATA PCENT /'10%','20%','30%','40%','50%','60%',
     +             '70%','80%','90%','95%','99%'/
       TCCOST  = 0
       TCMEAN  = 0
       TCVAR   = 0
       TCLMEAN = 0
       TCLVAR  = 0
       I       = 0
C      Compute Deciles
       WRITE (*,41)
       PRINT 2
       WRITE (*, '(1x,a\)') 'Enter Filename: '
       READ (*,'(A)') FLNM
C      Read Cost Data File
       OPEN (UNIT=4, FILE=FLNM(:LEN_TRIM(FLNM))//'.CST',
     +      STATUS='OLD',ERR=100)
       GOTO 110
100    CALL CLEARSCREEN( $GCLEARSCREEN )
       PRINT 1
       PRINT 2
       PRINT 2
       WRITE (*,48)
       PRINT 1
       PRINT 2
       PRINT 99
       READ (*,*)
       CALL MENU
110      DO WHILE (.NOT. EOF(4))
```

```
          I = I+1
          READ (4,21) COUNT(I),NAME(I),MIN(I),COST(I),MAX(I),DISTR(I),
     +                CORR(I),STDDEV(I)
            IF (DISTR(I) .EQ. 'U' .OR. DISTR(I) .EQ. 'u') THEN
C     Uniform Distribution
              CMEAN=(MIN(I)+MAX(I))/2
              CVAR(I)=((MAX(I)-MIN(I))**2)/12
            ELSE IF (DISTR(I) .EQ. 'T' .OR. DISTR(I) .EQ. 't') THEN
C     Triangular Distribution
              CMEAN=(MIN(I)+COST(I)+MAX(I))/3
              CVAR(I)=(MIN(I)*(MIN(I)-COST(I))+
     +             MAX(I)*(MAX(I)-MIN(I))+
     +             COST(I)*(COST(I)-MAX(I)))/18
            ELSE IF (DISTR(I) .EQ. 'N' .OR. DISTR(I) .EQ. 'n') THEN
C     Normal Distribution
              CMEAN=COST(I)
C     Insure Symmetry endpoints
                IF (ABS(MAX(I)-COST(I)) .GT. ABS(COST(I)-MIN(I))) THEN
                  MIN(I)=COST(I)-(MAX(I)-COST(I))
                ELSE
                  MAX(I)=(COST(I)-MIN(I))+COST(I)
                END IF
              CVAR(I)=((MAX(I)-MIN(I))/6)**2
            ELSE IF (DISTR(I) .EQ. 'L' .OR. DISTR(I) .EQ. 'l') THEN
C     Lognormal Distribution
              IF (STDDEV(I).EQ.0) THEN
                LMAX=LOG(MAX(I))
                  IF (MIN(I).LE.0) THEN
                    LMIN=0
                  ELSE
                    LMIN=LOG(MIN(I))
                  END IF
                LVAR=((LMAX-LMIN)/6)**2
                LMEAN=((LMAX-LMIN)/2)+LMIN
              ELSE
                LVAR=STDDEV(I)**2
                LMAX=LOG(MAX(I))
                LMIN=LOG(MIN(I))
                LMEAN=LOG(COST(I))+LVAR
              END IF
              CMEAN=EXP(LMEAN+(.5*LVAR))
              CVAR(I)=EXP((2*LMEAN)+LVAR)*(EXP(LVAR)-1)
            ELSE
C     BETA Distribution.
              CMEAN=(MIN(I)+4*COST(I)+MAX(I))/6
              CVAR(I)=((MAX(I)-MIN(I))**2)/36
            END IF
          TCMEAN=TCMEAN+CMEAN
          TCVAR=TCVAR+CVAR(I)
        END DO
C     Read Corrlation Matrix File
      OPEN (UNIT=5, FILE=FLNM(:LEN_TRIM(FLNM))//'.COR',
     +      STATUS='OLD',ERR=100)
        DO 120 L=1,I
          DO 130 M=1,I
```

```
          READ (5,5) CORMAT(L,M)
130         CONTINUE
120       CONTINUE
        CLOSE (UNIT=5,STATUS='KEEP')
C       Compute Covariance between elements
          DO 140 L=1,I
            DO 150 M=L+1,I
              TCVAR=TCVAR+2*CORMAT(L,M)*SQRT(CVAR(L))*SQRT(CVAR(M))
150         CONTINUE
140       CONTINUE
C       Log-Normal Computations
            TCLMEAN=.5*LOG(((TCMEAN**4)/((TCMEAN**2)+TCVAR)))
            TCLVAR=LOG(((TCMEAN**2)+TCVAR)/(TCMEAN**2))
        CALL CLEARSCREEN( $GCLEARSCREEN )
          DO 160 K=1,11
            PROB(K) = TCMEAN + ZSCORE(K) * SQRT(TCVAR)
            LPROB(K)= EXP(TCLMEAN+ZSCORE(K)*SQRT(TCLVAR))
160       CONTINUE
        CLOSE (UNIT=4,STATUS='KEEP')
200     CALL CLEARSCREEN( $GCLEARSCREEN )
        WRITE (*,24) FLNM(:LEN_TRIM(FLNM))
        PRINT 1
        WRITE (*,25) TCMEAN
        WRITE (*,26) SQRT(TCVAR)
        PRINT 1
        WRITE (*,27)
        WRITE (*,28)
          DO 210 K=1,11
            WRITE (*,29) PCENT(K), PROB(K), LPROB(K)
210       CONTINUE
        PRINT 1
        WRITE (*,40)
220     READ (*,*,ERR=200) SELECT
          IF (SELECT .EQ. 1) THEN
            PRINT 47
            READ (*,*,ERR=200) SELECT
              IF (SELECT .EQ. 3) THEN
                PRINT 42
                READ (*,*,ERR=200) SELECT
C       Normal Query
                  IF (SELECT .EQ. 5) THEN
                    PRINT 43
                    READ (*,*,ERR=200) EST
                    X = (EST - TCMEAN)/SQRT(TCVAR)
                    AREA = ALNORM(X) * 100
                    PRINT 45,AREA,'%'
                    READ (*,*)
                    GOTO 200
                  ELSE IF (SELECT .EQ. 6) THEN
                    PRINT 44
                    READ (*,*,ERR=200) P
                      IF (P.GT.1) THEN
                        P=P/100
                      ELSE IF (P.LT..0001) THEN
                        P=.0001
```

```
                    ELSE IF (P.GT..9999) THEN
                      P=.9999
                    END IF
                  Z = PPND16(P)
                  NEWCST = Z*SQRT(TCVAR)+TCMEAN
                  PRINT 46,NEWCST
                  READ (*,*)
                  GOTO 200
                ELSE
                  GOTO 200
                END IF
C     Lognormal Query
            ELSE IF (SELECT .EQ. 4) THEN
              PRINT 42
              READ (*,*,ERR=200) SELECT
                IF (SELECT .EQ. 5) THEN
                  PRINT 43
                  READ (*,*,ERR=200) EST
                  LEST = LOG(EST)
                  X = (LEST - TCLMEAN)/SQRT(TCLVAR)
                  AREA = ALNORM(X) * 100
                  PRINT 45,AREA,'%'
                  READ (*,*)
                  GOTO 200
                ELSE IF (SELECT .EQ. 6) THEN
                  PRINT 44
                  READ (*,*,ERR=200) P
                    IF (P.GE.1) THEN
                      P=P/100
                    ELSE IF (P.LT..0001) THEN
                      P=.0001
                    ELSE IF (P.GT..9999) THEN
                      P=.9999
                    END IF
                  Z = PPND16(P)
                  NEWCST = EXP(Z*SQRT(TCLVAR)+TCLMEAN)
                  PRINT 46,NEWCST
                  READ (*,*)
                  GOTO 200
                ELSE
                  GOTO 200
                END IF
            ELSE
              GOTO 200
            END IF
        ELSE IF (SELECT .EQ. 2) THEN
          CALL CLEARSCREEN( $GCLEARSCREEN )
          PRINT 2
          PRINT 2
          PRINT 2
          PRINT 2
          PRINT 2
          PRINT 2
          WRITE (*,50)
          READ (*,'(A)') CHOICE
```

```
         IF (CHOICE.EQ.'Y' .OR. CHOICE.EQ.'y') THEN
           OPEN (UNIT=2,FILE='LPT1')
           WRITE (2,1)
           WRITE (2,30) FLNM(:LEN_TRIM(FLNM))
           WRITE (2,31)
           WRITE (2,2)
           WRITE (2,2)
           WRITE (2,23)
           WRITE (2,1)
           WRITE (2,25) TCMEAN
           WRITE (2,26) SQRT(TCVAR)
           WRITE (2,1)
           WRITE (2,27)
           WRITE (2,28)
             DO 230 K=1,11
               WRITE (2,29) PCENT(K), PROB(K), LPROB(K)
230          CONTINUE
           WRITE (2,3)
           CLOSE (UNIT=2)
         ELSE
           CALL MENU
         END IF
       CALL MENU
     ELSE
       GOTO 200
     END IF
   READ (*,*)
   CALL CLEARSCREEN( $GCLEARSCREEN )
   CALL MENU
1  FORMAT (' ')
2  FORMAT ('0')
3  FORMAT ('1')
5  FORMAT (F5.2)
21 FORMAT (I3,A10,3F15.4,A1,I1,F8.5)
22 FORMAT (' ',F12.1)
23 FORMAT ('0',31X,'PROBABILITIES')
24 FORMAT ('0',31X,'PROBABILITIES',10X,A8,'.CST')
25 FORMAT (' ',20X,'Mean',15X,': ',F12.1)
26 FORMAT (' ',20X,'Standard Deviation : ',F12.1)
27 FORMAT (' ',32X,'Normal',12X,'Log Normal')
28 FORMAT (' ',12X,'Percentile',2(8X,'Value (est)'))
29 FORMAT (' ',16X,A3,8X,F12.1,7X,F12.1)
30 FORMAT ('0',25X,'COST UNCERTAINTY ANALYSIS',10X,A8,'.CST')
31 FORMAT (' ',31X,'Output Report')
40 FORMAT (' ',19X,'1) Query      2) Return to Main Menu  ',\)
41 FORMAT ('0',31X,'COST UNCERTAINTY')
42 FORMAT (' ',19X,'5) Cost -> Prob      6) Prob -> Cost  ',\)
43 FORMAT (' ',19X,'Enter Estimate  ',\)
44 FORMAT (' ',19X,'Enter Probability  ',\)
45 FORMAT (' ',32X,F6.2,A2)
46 FORMAT (' ',32X,F12.1)
47 FORMAT (' ',19X,'3) Normal              4) Lognormal  ',\)
48 FORMAT ('0',29X,'File does not exist')
50 FORMAT (' ',26X,'Print output report? (Y/N)  ',\)
99 FORMAT ('0',26X,'Press [RETURN] to Continue',\)
```

```
      END
C
C     Algorithm AS 66, Applied Statistics, 1973, Vol 22, No. 3
C
C     Evaluates the tail area of the standardized normal curve
C     from X to infinity if UPPER is .TRUE. or from minus infinity
C     to X if UPPER is .FALSE.
C
      REAL FUNCTION ALNORM(X)
      DOUBLE PRECISION X
      REAL LTONE, UTZERO, ZERO, HALF, ONE, CON, Y, Z
      LOGICAL UPPER /.FALSE./ ,UP
C
      DATA LTONE,UTZERO /7.0, 18.66/
      DATA ZERO, HALF, ONE, CON /0.0, 0.5, 1.0, 1.28/
      UP = UPPER
      Z = X
      IF (Z .GE. ZERO) GOTO 10
      UP = .NOT.UP
      Z = -Z
10    IF (Z.LE.LTONE .OR. UP .AND. Z.LE.UTZERO) GOTO 20
      ALNORM = ZERO
      GOTO 40
20    Y = HALF * Z * Z
      IF (Z .GT. CON) GOTO 30
C
      ALNORM = HALF-Z*(0.398942280385-0.399903438504*Y/
     1  (Y + 5.75885480458 - 29.8213557808 /
     2  (Y + 2.62433121679 + 48.6959930692 /
     3  (Y + 5.92885724438))))
      GOTO 40
C
30    ALNORM = 0.398942280385*EXP(-Y) /
     1  (Z - 3.8052E-8 + 1.00000615302 /
     2  (Z + 3.98064794E-4 + 1.98615381364 /
     3  (Z - 0.151679116635 + 5.29330324926 /
     4  (Z + 4.8385912808 - 15.1508972451 /
     5  (Z + 0.742380924027 + 30.789933034 / (Z + 3.99019417011))))))
C
40    IF (.NOT. UP) ALNORM = ONE - ALNORM
      RETURN
      END
C
C     Algorithm AS 241, Applied Statistics, 1988, Vol 37, No. 3
C
C     Produces the normal deviate 'Z' corresponding to a given
C     tail area of probability.  'Z' is accurate to approximately
C     1 part in 10^7.
C
      REAL FUNCTION PPND16(P)
      REAL ZERO, ONE, HALF, SPLIT1, SPLIT2, CONST1, CONST2,
     +  A0, A1, A2, A3, A4, A5, A6, A7, B1, B2, B3, B4, B5, B6, B7,
     +  C0, C1, C2, C3, C4, C5, C6, C7, D1, D2, D3, D4, D5, D6, D7,
     +  E0, E1, E2, E3, E4, E5, E6, E7, F1, F2, F3, F4, F5, F6, F7,
```

```
     +  P, Q, R
      PARAMETER (ZERO = 0.0E0, ONE = 1.0E0, HALF = ONE/2.0E0,
     +  SPLIT1 = .425E0,      SPLIT2 = 5.0E0,
     +  CONST1 = 0.180625E0, CONST2 = 1.6E0)
C
C     Coefficients for P close to 1/2
      PARAMETER (A0 = 3.38713 28727 96366 6080E0,
     +           A1 = 1.33141 66789 17843 7745E2,
     +           A2 = 1.97159 09503 06551 4427E3,
     +           A3 = 1.37316 93765 50946 1125E4,
     +           A4 = 4.59219 53931 54987 1457E4,
     +           A5 = 6.72657 70927 00870 0853E4,
     +           A6 = 3.34305 75583 58812 8105E4,
     +           A7 = 2.50908 09287 30122 6727E3,
     +           B1 = 4.23133 30701 60091 1252E1,
     +           B2 = 6.87187 00749 20579 0830E2,
     +           B3 = 5.39419 60214 24751 1077E3,
     +           B4 = 2.12137 94301 58659 5867E4,
     +           B5 = 3.93078 95800 09271 0610E4,
     +           B6 = 2.87290 85735 72194 2674E4,
     +           B7 = 5.22649 52788 52854 5610E3)
C     Hash sum AB    55.88319 28806 14901 4439
C
C     Coefficients for P neither close to 1/2 nor 0 or 1
      PARAMETER (C0 = 1.42343 71107 49683 57734E0,
     +           C1 = 4.63033 78461 56545 29590E0,
     +           C2 = 5.76949 72214 60691 40550E0,
     +           C3 = 3.64784 83247 63204 60504E0,
     +           C4 = 1.27045 82524 52368 38258E0,
     +           C5 = 2.41780 72517 74506 11770E-1,
     +           C6 = 2.27238 44989 26918 45833E-2,
     +           C7 = 7.74545 01427 83414 07640E-4,
     +           D1 = 2.05319 16266 37758 82187E0,
     +           D2 = 1.67638 48301 83803 84940E0,
     +           D3 = 6.89767 33498 51000 04550E-1,
     +           D4 = 1.48103 97642 74800 74590E-1,
     +           D5 = 1.51986 66563 61645 71966E-2,
     +           D6 = 5.47593 80849 95344 94600E-4,
     +           D7 = 1.05075 00716 44416 84324E-9)
C     Hash sum CD    49.33206 50330 16102 89036
C
C     Coefficients for P near 0 or 1
      PARAMETER (E0 = 6.65790 46435 01103 77720E0,
     +           E1 = 5.46378 49111 64114 36990E0,
     +           E2 = 1.78482 65399 17291 33580E01,
     +           E3 = 2.96560 57182 85048 91230E-1,
     +           E4 = 2.65321 89526 57612 30930E-2,
     +           E5 = 1.24266 09473 88078 43860E-3,
     +           E6 = 2.71155 55687 43487 57815E-5,
     +           E7 = 2.01033 43992 92288 13265E-7,
     +           F1 = 5.99832 20655 58879 37690E-1,
     +           F2 = 1.36929 88092 27358 05310E-1,
     +           F3 = 1.48753 61290 85061 48525E-2,
     +           F4 = 7.86869 13114 56132 59100E-4,
     +           F5 = 1.84631 83175 10054 68180E-5,
```

```
     +             F6 = 1.42151 17583 16445 88870E-7,
     +             F7 = 2.04426 31033 89939 78564E-15)
C     Hash sum EF    47.52583 31754 92896 71629
C
      IFAULT = 0
      Q = P - HALF
      IF (ABS(Q) .LE. SPLIT1) THEN
        R = CONST1 -Q * Q
        PPND16 = Q * (((((((A7 * R + A6) * R + A5) * R + A4) * R + A3)
     +     * R + A2) * R +A1) * R + A0) / (((((((B7 * R + B6) * R + B5)
     +     * R + B4) * R + B3) * R + B2) * R + B1) * R + ONE)
        RETURN
      ELSE
        IF (Q .LT. 0) THEN
          R = P
        ELSE
          R = ONE - P
        END IF
        IF (R .LE. ZERO) THEN
          IFAULT = 1
          PPND16 = ZERO
          RETURN
        END IF
        R = SQRT(-LOG(R))
        IF (R .LE. SPLIT2) THEN
          R = R - CONST2
        PPND16 = (((((((C7 * R + C6) * R + C5) * R + C4) * R
     +      + C3) * R + C2) * R + C1) * R + C0) / (((((((D7 * R
     +      + D6) * R + D5) * R + D4) * R + D3) * R + D2) * R
     +      + D1) * R + ONE)
        ELSE
          R = R - SPLIT2
        PPND16 = (((((((E7 * R + E6) * R + E5) * R + E4) * R
     +      + E3) * R + E2) * R + E1) * R + E0) / (((((((F7 * R
     +      + F6) * R + F5) * R + F4) * R + F3) * R + F2) * R
     +      + F1) * R + ONE)
        END IF
        IF (Q .LT. 0) PPND16 = -PPND16
        RETURN
      END IF
      END
C
C     Jacobi Subroutine
C
      SUBROUTINE jacobi(a,n,np,d,v,nrot)
C     a    = Name of Matrix
C     n    = Logical array dimension
C     np   = Size of Matrix (largest value)
C     d    = Eigenvalues
C     v    = Eigenvectors
C     nrot = Number of rotations
      INTEGER n,np,nrot,NMAX
      REAL a(np,np),d(np),v(np,np)
      PARAMETER (NMAX=500)
      INTEGER i,ip,iq,j
```

```
      REAL c,g,h,s,sm,t,tau,theta,tresh,b(NMAX),z(NMAX)
      do 12 ip=1,n
        do 11 iq=1,n
          v(ip,iq)=0.
11      continue
        v(ip,ip)=1.
12    continue
      do 13 ip=1,n
        b(ip)=a(ip,ip)
        d(ip)=b(ip)
        z(ip)=0.
13    continue
      nrot=0
      do 24 i=1,50
        sm=0.
        do 15 ip=1,n-1
          do 14 iq=ip+1,n
            sm=sm+abs(a(ip,iq))
14        continue
15      continue
        if(sm.eq.0.)return
        if(i.lt.4)then
          tresh=0.2*sm/n**2
        else
          tresh=0.
        endif
        do 22 ip=1,n-1
          do 21 iq=ip+1,n
            g=100.*abs(a(ip,iq))
            if((i.gt.4).and.(abs(d(ip))+
     *g.eq.abs(d(ip))).and.(abs(d(iq))+g.eq.abs(d(iq))))then
              a(ip,iq)=0.
            else if(abs(a(ip,iq)).gt.tresh)then
              h=d(iq)-d(ip)
              if(abs(h)+g.eq.abs(h))then
                t=a(ip,iq)/h
              else
                theta=0.5*h/a(ip,iq)
                t=1./(abs(theta)+sqrt(1.+theta**2))
                if(theta.lt.0.)t=-t
              endif
              c=1./sqrt(1+t**2)
              s=t*c
              tau=s/(1.+c)
              h=t*a(ip,iq)
              z(ip)=z(ip)-h
              z(iq)=z(iq)+h
              d(ip)=d(ip)-h
              d(iq)=d(iq)+h
              a(ip,iq)=0.
              do 16 j=1,ip-1
                g=a(j,ip)
                h=a(j,iq)
                a(j,ip)=g-s*(h+g*tau)
                a(j,iq)=h+s*(g-h*tau)
```

```
16              continue
                do 17 j=ip+1,iq-1
                  g=a(ip,j)
                  h=a(j,iq)
                  a(ip,j)=g-s*(h+g*tau)
                  a(j,iq)=h+s*(g-h*tau)
17              continue
                do 18 j=iq+1,n
                  g=a(ip,j)
                  h=a(iq,j)
                  a(ip,j)=g-s*(h+g*tau)
                  a(iq,j)=h+s*(g-h*tau)
18              continue
                do 19 j=1,n
                  g=v(j,ip)
                  h=v(j,iq)
                  v(j,ip)=g-s*(h+g*tau)
                  v(j,iq)=h+s*(g-h*tau)
19              continue
                nrot=nrot+1
              endif
21          continue
22        continue
          do 23 ip=1,n
            b(ip)=b(ip)+z(ip)
            d(ip)=b(ip)
            z(ip)=0.
23        continue
24      continue
        pause 'too many iterations in jacobi'
        return
        END
C   (C) Copr. 1986-92 Numerical Recipes Software ]*7KD[0p-031..
C
C       Program Termination
C
        SUBROUTINE TERM
        WRITE (*,1)
        WRITE (*,1)
        WRITE (*,1)
        WRITE (*,1)
        WRITE (*,1)
        WRITE (*,1)
1       FORMAT ('0')
        STOP
        END
```

Appendix C: Test Cases

The Automated Cost Uncertainty Program

TEST CASES

by

Dale N. Fletcher
ASA(RD&A)

The Automated Cost Uncertainty Program is a computer based analytical technique which can be used to perform cost uncertainty analysis. This program, in contrast to other simulated approaches such as Monte Carlo simulation, offers analysts the advantage of a quick response. Instead of waiting for the required number of iterations to be run on a computer, this program returns results instantaneously.

As recent graduates of the Airforce's Cost Analysis Program your expertise in the fields of risk and uncertainty analysis would be invaluable in ascertaining the useability, accurateness and usefulness of this program. It would therefore be of immeasurable benefit if you would take a few minutes and participate in this evaluation.

The testing procedure is as follows: two test cases have been devised which will examine various aspects of the program. You are asked to enter the data from the test cases into the program and obtain the results. Upon completion of the test, fill out the accompanying evaluation form. This form will be used to implement suggested improvements and rectify any shortcomings. Included in the package are the test cases (including correct answers), evaluation form, user's guide and a short participative demonstration of the program's editing functions.

Thank you for your participation.

Dale N. Fletcher

TEST CASE 1

Cost Data

	Min	ML	Max	Mean	Distribution
PAR	40	55	100	65.0	Triangular
ICS	20	30	65	38.3	Triangular
SIU	35	45	70	50.0	Triangular
SPM	30	40	50	40.0	Triangular
Data	10	15	20	15.0	Triangular
ST&E	5	10	25	13.3	Triangular

Correlation Matrix

	PAR	ICS	SIU	SPM	Data	ST&E
PAR	1					
ICS	0.8	1				
SIU	0.7	0.8	1			
SPM	0.5	0.6	0.7	1		
Data			0.4		1	
ST&E			0	0.6		1

COST UNCERTAINTY ANALYSIS   case1.CST
Output Report

PROBABILITIES

Mean : 221.7
Standard Deviation : 31.1

Percentile	Normal Value (est)	Log Normal Value (est)
10%	181.9	183.6
20%	195.5	195.2
30%	205.4	204.0
40%	213.8	211.9
50%	221.7	219.5
60%	229.5	227.4
70%	238.0	236.2
80%	247.8	246.9
90%	261.5	262.5
95%	272.7	276.1
99%	293.9	303.6

TEST CASE 2

COST DATA

	Cost Cell	Min	ML	Max	Std Dev	Distribution
1	Devel Eng	11,800	12,000	14,400	NA	Uniform
2	Prod, Eng, & Plan	3,800	4,200	5,500	NA	Triang
3	Tooling	NA	590	NA	0.159	Lognormal
4	Prototype Mfg	900	1,100	1,300	NA	Uniform
5	Data	340	400	460	NA	Beta
6	Sys Test & Eval	540	600	720	NA	Triang
7	Sys/Project Mgt	1,300	1,500	1,700	NA	Beta
8	Training	130	150	170	NA	Beta
9	Other	3,200	3,700	4,200	NA	Normal

CORRELATION MATRIX

	1	2	3	4	5	6	7	8	9
1	1								
2		1							
3		0.18	1						
4				1					
5		0.44	0.18		1				
6						1			
7		0.44	0.18				1		
8								1	
9		0.44	0.18						1

COST UNCERTAINTY ANALYSIS
Output Report

case2.CST

PROBABILITIES

Mean : 25682.8
Standard Deviation : 924.5

Percentile	Normal Value (est)	Log Normal Value (est)
10%	24498.0	24509.4
20%	24904.7	24900.5
30%	25198.0	25186.4
40%	25448.6	25433.3
50%	25682.8	25666.2
60%	25917.0	25901.2
70%	26167.6	26155.1
80%	26460.9	26455.4
90%	26867.6	26877.5
95%	27203.4	27231.2
99%	27833.5	27907.2

COST DATA workfile.CST

	NAME	MIN	MODE	MAX	Distribution
1	Dev Eng	11800.0	11999.9	14400.0	u
2	PEP	3800.0	4200.0	5500.0	t
3	Tooling	675.0	750.0	975.0	t
4	Prot Mfg	900.0	1100.0	1300.0	u
5	Data	340.0	400.0	460.0	b
6	SPM	1300.0	1500.0	1700.0	b
7	Init Trng	130.0	150.0	170.0	b
8	Facilities	720.0	900.0	1080.0	t
9	Other	3200.0	3500.0	4200.0	b
10	Investment	10700.0	11900.0	14300.0	t
11	Production	50500.0	56100.0	67300.0	t
12	ECO	1450.0	1450.0	2130.0	u
13	Sys T&E	480.0	600.0	720.0	t
14	OATA	990.0	1100.0	1210.0	b
15	Software	1.0	2.0	3.0	u
16	Training	72.0	80.0	88.0	t
17	Init Sps	3100.0	3400.0	4100.0	u
18	Transp	3000.0	3300.0	4000.0	b
19	MPA	309191.0	309300.0	338600.0	b
20	Repln Sps	27500.0	32400.0	40500.0	u
21	Dpt Mnt	28100.0	31200.0	35900.0	b
22	Mods	4000.0	4400.0	5300.0	u
23	Ind Supt	79000.0	92900.0	111500.0	b

THE AUTOMATED
COST
UNCERTAINTY PROGRAM
Editing

Introduction

The program disk contains two files, *WORKFILE.CST* and *WORKFILE.COR*. These files will be used as temporary files for a demonstration of the program's editing functions. These files may be discarded at the conclusion of the demonstration

Begin Demonstration

From the *MAIN MENU* select *FILE MANAGEMENT*
From the *FILE MANAGEMENT MENU* select *EDIT EXISTING FILE*
Enter the name of the temporary file (*WORKFILE*)
Select *EDIT DATA FILE*

The screen will display the data for the file to be edited. This demonstration will guide the user through the process of editing data files.

Cost Data File

The cost mode for Development Engineering (#1 *Dev Eng*) is incorrect. The cost mode should be 12000.

2) *Facilities* (#8) is no longer being included, this element should be deleted.

3) A new cost element *Testing*, needs to be added following element #5 *Data*. *Testing* has a triangular distribution with a minimum value of 540, a cost estimate of 600, and a maximum value of 720. This element is uncorrelated (independent).

4) The distribution for *Software* (#15) is thought to be lognormal with a mode of 87.874 and a standard deviation (σ) of .2936. This element is uncorrelated.

5) Change Replenishment Spare Parts (#20 *Repln Sps*) to lognormal with a cost mode of 7.86, a minimum value of .905, and a maximum value of 1211.97. This element is uncorrelated.

6) Change the distribution for Initial Spare Parts (# 17 *Init Sps*) to normal with minimum value of 55, maximum value of 100, and cost mode of 145. This element is uncorrelated.

7) The mode of #17 should be 100 and the maximum 145.

8) Other Modifications (#22 *Mods*) is also normally distributed with a mode of 40 and a standard deviation of 1.5. This element is also uncorrelated.

The editing of the data file has been completed. Select RETURN to exit the editing screen. To save the changes just made, save the now modified file under its previous filename. DO NOT overwrite the oldfile. Save the file under *NEWFILE*.

Return to the *FILE MANAGEMENT MENU*. The data can be printed using the print option. Select *PRINT FILE* and enter the name of the modified file (*NEWFILE*).

Return to the *MAIN MENU* and select *COST UNCERTAINTY*. An analysis can now be performed on the modified file, enter *NEWFILE*). Your answer should coincide with the supplied output report, *ANSWER1*.

All of the cost elements were thought be independent, in fact some are indeed correlated (dependent).

<u>Correlation Data File</u>

Return to the *MAIN MENU* (an output report may be obtained by answering 'y' at the prompt). Select *FILE MANAGEMENT* and *EDIT COST DATA* using the file *NEWFILE*.

Element #6 *Testing*, is correlated with element #1 *Dev Eng*. To change this relationship, *MODIFY* element #6. Enter 'n' to all prompts except change correlation. Here, enter 'y'. The *EDIT CORRELATION MATRIX* screen will be displayed.

Add the following correlations:

```
 1) Element #6 correlated with #1, rho = .9
```
Completed editing element #6

```
 2) Element #11 correlated with #12, rho =  .8
 3) Element #11 correlated with #17, rho =  .7
 4) Element #11 correlated with #21, rho =  .5
 5) Element #11 correlated with #22, rho =  .5
```
Completed editing element #11

```
 6) Element #12 correlated with #17, rho =  .8
 7) Element #12 correlated with #21, rho = -.6
 8) Element #12 correlated with #22, rho =  .8
```
Completed editing element #12

```
 9) Element #17 correlated with #21, rho =  .7
10) Element #17 correlated with #22, rho =  .4
```
Completed editing element #17

```
11) Element #21 correlated with #22, rho =  .6
```
Completed editing element #21 Select RETURN to save the file.

The correlation matrix is inconsistent because of an incorrect value entered. The rho

value of -.6 for element 12 should be +.6. Therefore we should *MODIFY* element #12. Enter 'n' to all prompts on the *REPLACE* screen since the only change will be to the correlation file. From the *EDIT CORRELATION MATRIX* screen, enter 'y' to change the value for #21 from =-.6 to .6. No more changes are necessary therefore the file can be saved.

This file will replace the previous file so we may overwrite the oldfile. Return to the *MAIN MENU* and select *COST UNCERTAINTY*. The modified *NEWFILE* is re-computed and although the total cost mean has not changed, the total cost standard deviation has increased. This is due to our taking into account the dependencies between elements. The output report *ANSWER2*, displays these new values.

This demonstration was designed to familiarize the user with the flexibility of the editing functions.

THE AUTOMATED COST UNCERTAINTY PROGRAM
Evaluation Form

EASE OF USE:
Circle one

User Friendly	Yes	No
Program Clarity	Easy to follow	Difficult to follow
Speed of program	Speed OK	Too Slow
Information	Adequate	Require More

ERRORS
If program crashed what led up to error?

COMMENTS
Suggested Improvements:

Criticisms:

BIBLIOGRAPHY

1. Abramson, Robert L. and Philip H. Young. *FRISKEM - Formal Risk Evaluation Methodology*. Los Angeles CA: The Aerospace Corporation, 29-31 October, 1990.

2. Adams, A.G. "Areas Under the Normal Curve (Algorithm 39)," *Computer Journal, 12*: 197-198 1969.

3. Ageloff, Roy and Richard Mojena. *Applied FORTRAN 77 Featuring Structured Programming*. Belmont CA: Wadsworth Publishing Company, 1981.

4. Boguslavsky, Boris W. *Elementary Computer Programming in Fortran IV* (Second Edition). Reston VA: Reston Publishing Company, Inc., 1980.

5. Bronson, Richard. *Matrix Methods: An Introduction* (Second Edition). Boston MA: Academic Press, Inc., 1991.

6. Cooper, B.E. "The Normal Integral (Algorithm AS2)," *Applied Statistics, Journal of the Royal Statistical Society, 17*: 186-188 (1968).

7. Devore, Jay L. *Probability and Statistics for Engineering and the Sciences* (Third Edition). Pacific Grove CA: Brooks/Cole Publishing Company, 1991.

8. Dienemann, Paul F. *Estimating Cost Uncertainty Using Monte Carlo Techniques*. Santa Monica CA: RM-4854PR, The RAND Corporation, January 1966.

9. Dunnett, Charles W. "The Normal Integral (Algorithm AS251)," *Applied Statistics, Journal of the Royal Statistical Society, 38*: (1989).

10. Dwight, Herbert Bristol. *Tables of Integrals and Other Mathematical Data*. New York NY: The Macmillan Company, 1947.

11. Fisher, G.H. *A Discussion of Uncertainty in Cost Analysis*, The Rand Corporation, Report RM-3071-PR, April 1962.

12. Fisher, G.H. *The Nature of Uncertainty*, The Rand Corporation, Report P-5133, November 1973.

13. Freeman, Peter and Anthony I. Wasserman. *Tutorial on Software Design Techniques* (Fourth Edition). Los Angeles CA: IEEE Computer Society Press, 1983.

14. Garvey, Paul R. *A General Approach to System Cost Uncertainty Analysis.* Contract F19628-89-C-0001. Bedford MA: The MITRE Corporation, September 1989.

15. Garvey, Paul R. and Aurdrey T. Taub. *A Joint Probability Model for Cost and Schedule Uncertainties.* Presentation to the 26th Annual DODCAS, Sept 1992.

16. Garvey, Paul R. *Foundations of Cost Uncertainty Analysis, A Systems Engineering Perspective.* Draft Manuscript.

17. Gregory, Robert T. and David L. Karney. *A Collection of Matrices for Testing Computational Algorithms.* Huntington NY: Robert E. Krieger Publishing Company, 1978.

18. Hill, Ian D. and S. A. Joyce. "Normal Curve Integral (Algorithm 304)," *Communications of the Association for Computing Machinery, 10*: 374-375 (June 1967).

19. Hill, Ian D. "The Normal Integral (Algorithm AS 66)," *Applied Statistics, Journal of the Royal Statistical Society, 22*: 424-427 (1973).

20. Hutchison, Larry D. *A Microcomputer Program for the Solution of Learning Curve Computations.* MS thesis, AFIT/GSM/LSQ/85S-18. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1985 (AD-A161648).

21. Kazanowski, A.D. *A Quantitive Method for Estimating Total System Cost Risk*, Aerospace Corp, Report SD-TR-83-44, 3 June 1983.

22. Law, A.M. and D.W. Kelton. *Simulation Modeling and Analysis.* McGraw Hill, 1982, Chapter 5, pages 155-216.

23. Lurie, Philip M., Matthew S. Goldberg, and Mitchell S. Robinson. *A Handbook of Cost Risk Analysis Methods*, Institute for Defense Analysis, IDA Paper P-2734, April 1993.

24. Maron, Melvin J. and Robert J. Lopez. *Numerical Analysis A Practical Approach* (Third Edition). Belmont CA: Wadsworth Inc., 1991.

25. Murphy, R.L., W.P. Simpson, and T.R. O'Hara. *An Overview of Cost Risk Analysis*, Presentation to the 26th Annual DODCAS, Sept 1992.

26. O'Hara, Thomas R. *An Investigation of the Air Force Risk Model*. MS thesis, AFIT/GCA/LSQ/91S-9. School of Systems and Logistics, Air Force Institute of Technology (AU), Wright-Patterson AFB OH, September 1991 (AD-A243921).

27. Pierce, B.O. *A Short Table of Integrals*. Boston MA: Ginn and Company, 1929.

28. Powell, Patricia B. *Software Validation, Verification, and Testing Technique and Tool Reference Guide*. Washington DC: U.S. Department of Commerce, National Bureau of Standards, 1982.

29. Searle, Shayle R. *Matrix Algebra Useful for Statistics*. New York NY: John Wiley & Sons, 1982.

30. Simpson, Wendell P. and Kevin P. Grant. *An Investigation of the Accuracy of Heuristic Methods for Cost Uncertainty Analysis*. AU-AFIT/LA-TR-94-1. Air Force Institute of Technology/LAS, Wright-Patterson AFB OH, September 1991.

31. Smith, Brian T. *Matrix Eigensystem Routines. EISPACK Guide*. Berlin: Springer-Verlag, 1974.

32. U.S. Army Logistics Management Center. *Reference Book for Cost Estimating for Engineers Course (CEE)*, ALM-63-0219-RB(A), FT. Lee VA, 1984.

33. Vetterling, William T., Saul A. Teukolsky, William H. Press, and Brian P. Flannery. *Numerical Recipes Example Book (FORTRAN)*, Cambridge: Cambridge University Press, 1985.

34. Weist, J.D., and F.K. Levy. *A Management Guide to PERT/CPM* (Second Edition). Prentice-Hall, 1977.

35. Wichura, Michael J. "The Percentage Points of the Normal Distribution (Algorithm AS 241)," *Applied Statistics, Journal of the Royal Statistical Society, 35*: 477-484 (1988).

36. Wilkinson, J. H. *The Algebraic Eigenvalue Problem*. Oxford: Oxford Clarendon Press, 1965.

37. Young, Phillip H. *FRISK-Formal Risk Assessment of System Cost Estimates*. Los Angeles CA: AIAA 92-1054, The Aerospace Corporation, February 1992.

Vita

Mr. Dale N. Fletcher was born 19 March 1957 in Chicago, Illinois. He graduated from St. Francis de Sales High School in 1975. After completing his commitment with the United States Navy, he attended Chicago State University graduating with a Bachelor of Science degree with a major in Mathematics and a minor in Physics. Upon graduation, he worked for the Illinois Institute of Technology Research Institute. In October 1985 he entered Government service as an operations research systems analyst under first the Comptroller of the Army and then the Assistant Secretary of the Army (Financial Management) at the U.S. Army Cost and Economic Analysis Center. Prior to entering the School of Logistics and Acquisition Management in May 1993, he became a program analyst with the Assistant Secretary of the Army (Research, Development and Acquisition). He is married to the former Henrietta Denise Welters.

Permanent address: 9725 South Prairie Avenue
Chicago, Illinois 60628

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)

2. REPORT DATE
September, 1994

3. REPORT TYPE AND DATES COVERED
Master's Thesis

4. TITLE AND SUBTITLE
An Automated Cost Uncertainty Program

5. FUNDING NUMBERS

6. AUTHOR(S)
Dale N. Fletcher

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Institute of Technology, WPAFB OH 45433

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/GCA/LAS/94S-4

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

The purpose of this research was to develop a menu driven, computer program that implemented heuristic methodologies for cost uncertainty analysis. This research resulted in the development of an automated heuristic cost uncertainty program which alleviates the limitations of current computer programs. The program enables the analyst to compute probabilities associated with independent or dependent cost elements exhibiting either normal or lognormal total cost distributions. It consists of three modules. The first module contains a file editor which allows the user to create, modify and view existing cost and correlation data files. The second is a cost uncertainty module which is used for computing decile level total system cost probabilities. The last module contains a query function which enables the analyst to compute probabilities associated with different costs and the costs associated with different uncertainty levels. The program also performs a consistency check of the correlation matrix. The FORTRAN computer program is completely menu driven for ease of use.

14. SUBJECT TERMS
Cost Uncertainty, Cost Analysis, Probability

15. NUMBER OF PAGES
106

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL